issues of a dataset, and can help with generating further hypotheses. Chapter 16, “data analysis”, presents the theory and methods for model development (Sect. 16.1) as well as common data analysis techniques in clinical studies, namely linear regression (Sect. 16.2), logistic regression (Sect. 16.3) and survival analysis including Cox proportional hazards models (Sect. 16.4). Finally, Chap. 17 discusses the principles of model validation and sensitivity analyses, where the results of a particular research are tested for robustness in the face of varying model assumptions.

Each chapter includes worked examples inspired from a unique study, published in Chest in 2015 by Hsu et al., which addressed a key question in clinical practice in intensive care medicine: “is the placement of an indwelling arterial catheter (IAC) associated with reduced mortality, in patients who are mechanically ventilated but do not require vasopressor support?” IACs are used extensively in the intensive care unit for continuous monitoring of blood pressure and are thought to be more accurate and reliable than standard, non-invasive blood pressure monitoring. They also have the added benefit of allowing for easier arterial blood gas collection which can reduce the need for repeated arterial punctures. Given their invasive nature, however, IACs carry risks of bloodstream infection and vascular injury, so the evidence of a beneficial effect requires evaluation. The primary outcome of interest selected was 28-day mortality with secondary outcomes that included ICU and hospital length-of-stay, duration of mechanical ventilation, and mean number of blood gas measurements made. The authors identified the encounter-centric ‘arterial catheter placement’ as their exposure of interest and carried out a propensity score analysis to test the relationship between the exposure and outcomes using MIMIC. The result in this particular dataset (spoiler alert) is that the presence of an IAC is not associated with a difference in 28-day mortality, in hemodynamically stable patients who are mechanically ventilated. This case study provides a basic foundation to apply the above theory to a working example, and will give the reader first-hand perspective on various aspects of data mining and analytical techniques. This is in no way a comprehensive exploration of EHR analytics and, where the case lacks the necessary detail, we have attempted to include additional relevant information for common analytical techniques. For the interested reader, references are provided for more detailed readings.

Chapter 9
Formulating the Research Question

Anuj Mehta, Brian Malley and Allan Walkey

Learning Objectives

- Understand how to turn a clinical question into a research question.
- Principles of choosing a sample.
- Approaches and potential pitfalls.
- Principles of defining the exposure of interest.
- Principles of defining the outcome.
- Selecting an appropriate study design.

9.1 Introduction

The clinical question arising at the time of most health-care decisions is: "will this help my patient?" Before embarking on an investigation to provide data that may be used to inform the clinical question, the question must be modified into a research query. The process of developing a research question involves defining several components of the study and also what type of study is most suited to utilize these components to yield valid and reliable results. These components include: in whom is this research question relevant? The population of subjects defined by the researcher is referred to as the sample. The drug, maneuver, event or characteristic that we are basing our alternative hypothesis on is called the exposure of interest. Finally, the outcome of interest must be defined. With these components in mind the researcher must decide which study design is best or most feasible for answering the question. If an observational study design is chosen, then the choice of a database is also crucial.

In this chapter, we will explore how researchers might work through converting a clinical question into a research question using the clinical scenario of indwelling

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_9

arterial catheters (IAC) use during mechanical ventilation (MV). Furthermore, we will discuss the strengths and weaknesses of common study designs including randomized controlled trials as well as observational studies.

9.2 The Clinical Scenario: Impact of Indwelling Arterial Catheters

Patients who require MV because they are unable to maintain adequate breathing on their own (e.g. from severe pneumonia or asthma attack) are often the sickest patients in the hospital, with mortality rates exceeding 30 % [1–3]. Multiple options are available to monitor the adequacy of respiratory support for critically ill patients requiring MV, ranging from non-invasive trans-cutaneous measures to invasive, indwelling monitoring systems. IACs are invasive monitoring devices that allow continuous real time blood pressure monitoring and facilitate access to arterial blood sampling to assess arterial blood pH, oxygen and carbon dioxide levels, among others [4–6]. While closer monitoring of patients requiring MV with IACs may appear at face value to be beneficial, IACs may result in severe adverse events, including loss of blood flow to the hand and infection [7, 8]. Currently, data is lacking whether benefits may outweigh risks of more intensive monitoring using IACs. Examining factors associated with the decision to use IACs, and outcomes in patients provided IACs as compared to non-invasive monitors alone, may provide information useful to clinicians facing the decision as to whether to place an IAC.

9.3 Turning Clinical Questions into Research Questions

The first step in the process of transforming a clinical question into research is to carefully define the **study sample (or patient cohort)**, the **exposure** of interest, and the **outcome** of interest. These 3 components—sample, exposure, and outcome—are essential parts of every research question. Slight variations in each component can dramatically affect the conclusions that can be drawn from any research study, and whether the research will appropriately address the overarching clinical question.

9.3.1 Study Sample

In the case of IAC use, one might imagine many potential study samples of interest: for example, one might include all ICU patients, all patients receiving MV, all patients receiving intravenous medications that strongly affect blood pressure, adults only, children only, etc. Alternatively, one could define samples based on specific diseases or syndrome, such as shock (where IACs may be used to closely monitor blood pressure) or severe asthma (where IAC may be used to monitor oxygen or carbon dioxide levels).

The choice of study sample will affect both the internal and the external validity (generalizability) of the study. A study focusing only on a pediatric population may not apply to the adult population. Similarly, a study focused on patients receiving MV may not be applicable to non-ventilated patients. Furthermore, a study including patients with different reasons for using an IAC, with different outcomes related to the reason for IAC use, may lack internal validity due to bias called 'confounding'. Confounding is a type of study bias in which an exposure variable is associated with both the exposure and the outcome.

For instance, if the benefits of IACs on mortality are studied in all patients receiving MV, researchers must take into account the fact that IAC placement may actually be indicative of greater severity of illness. For example, imagine a study with a sample of MV patients in which those with septic shock received an IAC to facilitate vasoactive medications and provide close blood pressuring monitoring while patients with asthma did not receive an IAC as other methods were used to monitor their ventilation (such as end-tidal CO_2 monitoring). Patients with septic shock tend to have a much higher severity of illness compared to patients with asthma regardless of whether an IAC is placed. In such a study, researchers may conclude that IACs are associated with higher mortality only because IACs were used in sicker patients with a higher risk of dying. The variable "diagnosis" is therefore a confounding factor, associated with both the exposure (decision to insert an IAC) and the outcome (death). Careful sample selection is one method of attempting to address issues of confounding related to severity of illness. Restricting study samples to exclude groups that may strongly confound results (i.e. no patients on vasoactive medications) is one strategy to reduce bias. However, the selection of homogeneous study samples to increase internal validity should be balanced with the desire to generalize study findings to broader patient populations. These principles are discussed more extensively in the Chap. 10—"Cohort Selection".

9.3.2 Exposure

The exposure in our research question appears to be fairly clear: placement of an IAC. However, careful attention should be paid as to how each exposure or variable of interest is defined. Misclassifying exposures may bias results. How should IAC be measured? For example, investigators may use methods ranging from direct review of the medical chart to use of administrative claims data (i.e. International Classification of Diseases—ICD-codes) to identify IAC use. Each method of ascertaining the exposure of interest may have pros (improved accuracy of medical chart review) and cons (many person-hours to perform manual chart review).

Defining the time window during which an exposure of interest is measured may also have substantial implications that must be considered when interpreting the research results. For the purposes of our IAC study, the presence of an IAC was defined as having an IAC placed after the initiation of MV. The time-dependent nature of the exposure is critical for answering the clinical question; some IACs placed prior to MV are for monitoring of low-risk surgical patients in the operating room. Including all patients with IACs regardless of timing may bias the results towards a benefit for IACs by including many otherwise healthy patients who had an IAC placed for surgical monitoring. Alternatively, if the exposure group is defined as patients who had an IAC at least 48 h after initiation of MV, the study is at risk for a type of confounding called "immortal time bias": only patients who were alive could have had an IAC placed, whereas patients dying prior to 48 h (supposedly sicker) could not have had an IAC.

Equally important to defining the group of patients who received or experienced an exposure is to define the "unexposed" or control group. While not all research requires a control group (e.g. epidemiologic studies), a control group is needed to assess the effectiveness of healthcare interventions. In the case of the IAC study, the control group is fairly straightforward: patients receiving MV who did not have an IAC placed. However, there are important nuances when defining control groups. In our study example, an alternate control group could be all ICU patients who did not receive an IAC. However, the inclusion of patients not receiving MV results in a control group with a lower severity of illness and expected mortality than patients receiving MV, which would bias in favor of not using IACs. Careful definition of the control group is needed to properly interpret any conclusions from research; defining an appropriate control group is as important as defining the exposure.

9.3.3 Outcome

Finally, the investigator needs to determine the outcome of interest. Several different types of outcomes can be considered, including intermediate or mechanistic outcomes (informs etiological pathways, but may not immediately impact patients), patient-centered outcomes (informs outcomes important to patients, but may lack mechanistic insights: e.g. comfort scales, quality of life indices, or mortality), or healthcare-system centered outcomes (e.g. resource utilization, or costs). In our example of IAC use, several outcomes could be considered including intermediate outcomes (e.g. number of arterial blood draws, ventilator setting changes, or vasoactive medication changes), patient-centered outcomes (e.g. 28-day or 90-day mortality, adverse event rates), or healthcare utilization (e.g. hospitalization costs, added clinician workload). As shown in our example, outcome(s) may build upon each other to yield a constellation of findings that provides a more complete picture to address the clinical question of interest.

After clearly defining the study sample, exposure of interest, and outcome of interest, a research question can be formulated. A research question using our example may be formulated as follows:

"*In the population of interest (**study cohort**), is the exposure to the **variable of interest** associated with a different **outcome** than in the **control group**?*", which becomes, in our example:

"*Among mechanically ventilated, adult ICU patients who are not receiving vasoactive medications (i.e., the study sample) is placement of an IAC after initiation of MV (as compared with not receiving an IAC) (i.e. the exposure and control patients) associated with improved 28-day mortality rates (primary outcome, patient-centered) and the number of blood gas measurements per day (supporting secondary outcome, intermediate/mechanistic)?*"

9.4 Matching Study Design to the Research Question

Once the research question has been defined, the next step is to choose the optimal study design given the question and resources available. In biomedical research, the gold-standard for study design remains the double-blinded, randomized, placebo-controlled trial (RCT) [9, 10]. In a RCT, patients with a given condition (e.g. all adults receiving MV) would be randomized to receive a drug or intervention of interest (e.g. IAC) or randomized to receive the control (e.g. no IAC), with careful measurement of pre-determined outcomes (e.g. 28-day mortality). In ideal conditions, the randomization process eliminates all measured and unmeasured confounding and allows for causal inferences to be drawn, which cannot generally be achieved without randomization. As shown above, confounding is a threat to valid inferences from study results. Alternatively, in our example of septic shock verses asthma, severity of illness associated with the underlying condition may represent another confounder. Randomization solely based on the exposure of interest attempts to suppress issues of confounding. In our examples, proper randomization in a large sample would theoretically create equal age distributions and equal numbers of patients with septic shock and asthma in both the exposure and the control group.

However, RCTs have several limitations. Although the theoretical underpinnings of RCTs are fairly simple, the complex logistics of patient enrollment and retention, informed consent, randomization, follow up, and blinding may result in RCTs deviating from the 'ideal conditions' necessary for unbiased, causal inference. Additionally, RCTs carry the highest potential for patient harm and require intensive monitoring because the study dictates what type of treatment a patient receives (rather than the doctor) and may deviate from routine care. Given the logistic complexity, RCTs are often time- and cost-intensive, frequently taking many years and millions of dollars to complete. Even when logistically feasible, RCTs often 'weed out' multiple groups of patients in order to minimize potential harms and maximize detection of associations between interventions and outcomes of interest. As a result, RCTs can consist of homogeneous patients meeting narrow criteria, which may reduce the external validity of the studies' findings. Despite much effort and cost, an RCT may miss relevance to the clinical question as to whether the intervention of interest is helpful for your particular patient or not. Finally, some clinical questions may not ethically be answered with RCTs. For instance, the link between smoking and lung cancer has never been shown in a RCT, as it is unethical to randomize patients to start smoking in a smoking intervention group, or randomize patients to a control group in a trial to investigate the efficacy of parachutes [11]!

Observational research differs from RCTs. Observational studies are non-experimental; researchers record routine medical practice patterns and derive conclusions based on correlations and associations without active interventions [9, 12]. Observational studies can be retrospective (based on data that has already been collected), prospective (data is actively collected over time), or ambi-directional (a mix). Unlike RCTs, researchers in observational studies have no role in deciding what types of treatments or interventions patients receive. Observational studies tend to be logistically less complicated than RCTs as there is no active intervention, no randomization, no data monitoring boards, and data is often collected retrospectively. As such, observational studies carry less risk of harm to patients (other than loss of confidentiality of data that has been collected) than RCTs, and tend to be less time- and cost-intensive. Retrospective databases like MIMIC-II [13] or the National Inpatient Sample [14] can also provide much larger study samples (tens of thousands in some instances) than could be enrolled in an RCT, thus providing larger statistical power. Additionally, broader study samples are often included in observational studies, leading to greater generalizability of the results to a wider range of patients (external validity). Finally, certain clinical questions that would be unethical to study in an RCT can be investigated with observational studies. For example, the link between lung cancer and tobacco use has been demonstrated with multiple large prospective epidemiological studies [15, 16] and the life-saving effects of parachutes have been demonstrated mostly through the powers of observation.

Although logistically simpler than RCTs, the theoretical underpinnings of observational studies are generally more complex than RCTs. Obtaining causal estimates of the effect of a specific exposure on a specific outcome depends on the philosophical concept of the ‘counterfactual’ [17]. The counterfactual is the situation in which, all being equal, the same research subject at the same time would receive the exposure of interest and (the counterfactual) not receive the exposure of interest, with the same outcome measured in the exposed and unexposed research subject. Because we cannot create cloned research subjects in the real-world, we rely on creating groups of patients similar to the group that receives an intervention of interest. In the case of an ideal RCT with a large enough number of subjects, the randomization process used to select the intervention and control groups creates two alternate ‘universes’ of patients that will be similar except as related to the exposure of interest. Because observational studies cannot intervene on study subjects, observational studies create natural experiments in which the counterfactual group is defined by the investigator and by clinical processes occurring in the real-world. Importantly, real-world clinical processes often occur for a reason, and these reasons can cause deviation from counterfactual ideals in which exposed and unexposed study subjects differ in important ways. In short, observational studies may be more prone to bias (problems with internal validity) than RCTs due to difficulty obtaining the counterfactual control group.

Several types of biases have been identified in observational studies. Selection bias occurs when the process of selecting exposed and unexposed patients introduces a bias into the study. For example, the time between starting MV and receiving IAC may introduce a type of "survivor treatment selection bias" since patients who received IAC could not have died prior to receiving IACs. Information bias stems from mismeasurement or misclassification of certain variables. For retrospective studies, the data has already been collected and sometimes it is difficult to evaluate for errors in the data. Another major bias in observational studies is confounding. As stated, confounding occurs when a third variable is correlated with both the exposure and outcome. If the third variable is not taken into consideration, a spurious relationship between the exposure and outcome may be inferred. For example, smoking is an important confounder in several observational studies as it is associated with several other behaviors such as coffee and alcohol consumption. A study investigating the relationship between coffee consumption and incidence of lung cancer may conclude that individuals who drink more coffee have higher rates of lung cancer. However, as smoking is associated with both coffee consumption and lung cancer, it is confounder in the relationship between coffee consumption and lung cancer if unmeasured and unaccounted for in analysis. Several methods have been developed to attempt to address confounding in observational research such as adjusting for the confounder in regression equations if it is known and measured, matching cohorts by known confounders, and using instrumental variables—methods that will be explained in-depth in future chapters. Alternatively, one can restrict the study sample (e.g. excluding patients with shock from a study evaluating the utility of IACs). For these reasons, while powerful, an individual observational study can, at best, demonstrate associations and correlations and cannot prove causation. Over time, a cumulative sum of multiple high quality observational studies coupled with other mechanistic evidence can lead to causal conclusions, such as in the causal link currently accepted between smoking and lung cancer established by observational human studies and experimental trials in animals.

9.5 Types of Observational Research

There are multiple different types of questions that can be answered with observational research (Table 9.1). Epidemiological studies are one major type of observational research that focuses on the burden of disease in predefined populations. These types of studies often attempt to define incidence, prevalence, and risk factors for disease. Additionally, epidemiological studies also can investigate changes to healthcare or diseases over time. Epidemiological studies are the cornerstone of public health and can heavily influence policy decisions, resource

Table 9.1 Major types of observational research, and their purpose

Type of observational research	Purpose
Epidemiological	Define incidence, prevalence, and risk factors for disease
Predictive modeling	Predict future outcomes
Comparative effectiveness	Identify intervention associated with superior outcomes
Pharmacovigilance	Detect rare drug adverse events occurring in the long-term

allocation, and patient care. In the case of lung cancer, predefined groups of patients without lung cancer were monitored for years until some patients developed lung cancer. Researchers then compared numerous risk factors, like smoking, between those who did and did not develop lung cancer which led to the conclusion that smoking increased the risk of lung cancer [15, 16].

There are other types of epidemiological studies that are based on similar principles of observational research but differ in the types of questions posed. Predictive modeling studies develop models that are able to accurately predict future outcomes in specific groups of patients. In predictive studies, researchers define an outcome of interest (e.g. hospital mortality) and use data collected on patients such as labs, vital signs, and disease states to determine which factors contributed to the outcome. Researchers then validate the models developed from one group of patients in a separate group of patients. Predictive modeling studies developed many common prediction scores used in clinical practice such as the Framingham Cardiovascular Risk Score [18], APACHE IV [19], SAPS II [20], and SOFA [21].

Comparative effectiveness research is another form of observational research which involves the comparison of existing healthcare interventions in order to determine effective methods to deliver healthcare. Unlike descriptive epidemiologic studies, comparative effectiveness research compares outcomes between similar patients who received different treatments in order to assess which intervention may be associated with superior outcomes in real-world conditions. This could involve comparing drug A to drug B or could involve comparing one intervention to a control group who did not receive that intervention. Given that there are often underlying reasons why one patient received treatment A versus B or an intervention versus no intervention, comparative effectiveness studies must meticulously account for potential confounding factors. In the case of IACs, the research question comparing patients who had an IAC placed to those who did not have an IAC placed would represent a comparative effectiveness study.

Pharmacovigilance studies are yet another form of observational research. As many drug and device trials end after 1 or 2 years, observational methods are used to evaluate if there are patterns of rarer adverse events occurring in the long-term. Phase IV clinical studies are one form of pharmacovigilance studies in which long-term information related to efficacy and harm are gathered after the drug has been approved.

9.6 Choosing the Right Database

A critical part of the research process is deciding what types of data are needed to answer the research question. Administrative/claims data, secondary use of clinical trial data, prospective epidemiologic studies, and electronic health record (EHR) systems (both from individual institutions and those pooled from multiple institutions) are several sources from which databases can be built. Administrative or claims databases, such as the National Inpatient Sample and State Inpatient Databases complied by the Healthcare Cost and Utilization Project or the Medicare database, contain information on patient and hospital demographics as well as billing and procedure codes. Several techniques have been developed to translate these billing and procedure codes to more clinically useful disease descriptions. Administrative databases tend to provide very large sample sizes and, in some cases, can be representative of an entire population. However, they lack granular patient-level data from the hospitalization such as vital signs, laboratory and microbiology data, timing data (such as duration of MV or days with an IAC) or pharmacology data, which are often important in dealing with possible confounders.

Another common source of data for observational research is large epidemiologic studies like the Framingham Heart Study as well as large multicenter RCTs such as the NIH ARDS Network. Data that has already been can be analyzed retrospectively with new research questions in mind. As the original data was collected for research purposes, these types of databases often have detailed, granular information not available in other clinical databases. However, researchers are often bound by the scope of data collection from the original research study which limits the questions that may be posed. Importantly, generalizability may be limited in data from trials.

The advent of Electronic Health Records (EHR) has resulted in the digitization of medical records from their prior paper format. The resulting digitized medical records present opportunities to overcome some of the shortcomings of administrative data, yielding granular data with laboratory results, medications, and timing of clinical events [13]. These “big databases” take advantage of the fact many EHRs collect data from a variety of sources such as patient monitors, laboratory systems, and pharmacy systems and coalesce them into one system for clinicians. This information can then be translated into de-identified databases for research purposes that contain detailed patient demographics, billing and procedure information, timing data, hospital outcomes data, as well as patient-level granular data and provider notes which can searched using natural language processing tools. “Big data” approaches may attenuate confounding by providing detailed information needed to assess severity of illness (such as lab results and vital signs). Furthermore, the granular nature of the data can provide insight as to the reason why one patient received an intervention and another did not which can partly address confounding by indication. Thus, the promise of “big data” is that it contains small, very detailed data. “Big data” databases, such as MIMIC-III, have the potential to expand the scope of what had previously been possible with observational research.

9.7 Putting It Together

Fewer than 10 % of clinical decisions are supported by high level evidence [22]. Clinical questions arise approximately in every other patient [23] and provide a large cache of research questions. When formulating a research question, investigators must carefully select the appropriate sample of subjects, exposure variable, outcome variable, and confounding variables. Once the research question is clear, study design becomes the next pivotal step. While RCTs are the gold standard for establishing causal inference under ideal conditions, they are not always practical, cost-effective, ethical or even possible for some types of questions. Observational research presents an alternative to performing RCTs, but is often limited in causal inference by unmeasured confounding.

Our clinical scenario gave rise to the question of whether IACs improved the outcomes of patients receiving MV. This translated into the research question: "Among mechanically ventilated ICU patients not receiving vasoactive medications (study sample) is use of an IAC after initiation of MV (exposure) associated with improved 28-day mortality (outcome)?" While an RCT could answer this question, it would be logistically complex, costly, and difficult. Using comparative effectiveness techniques, one can pose the question using a granular retrospective database comparing patients who received an IAC to measurably similar patients who did not have an IAC placed. However, careful attention must be paid to unmeasured confounding by indication as to why some patients received IAC and others did not. Factors such as severity of illness, etiology of respiratory failure, and presence of certain diseases that make IAC placement difficult (such as peripheral arterial disease) may be considered as possible confounders of the association between IAC and mortality. While an administrative database could be used, it could lack important information related to possible confounders. As such, EHR databases like MIMIC-III, with detailed granular patient-level data, may allow for measurement of a greater number of previously unmeasured confounding variables and allow for greater attenuation of bias in observational research.

Take Home Messages

- Most research questions arise from clinical scenarios in which the proper course of treatment is unclear or unknown.
- Defining a research question requires careful consideration of the optimal study sample, exposure, and outcome in order to answer a clinical question of interest.
- While observational research studies can overcome many of the limitations of randomized controlled trials, careful consideration of study design and database selection is needed to address bias and confounding.

References

1. Esteban A, Frutos-Vivar F, Muriel A, Ferguson ND, Peñuelas O, Abraira V, Raymondos K, Rios F, Nin N, Apezteguía C, Violi DA, Thille AW, Brochard L, González M, Villagomez AJ, Hurtado J, Davies AR, Du B, Maggiore SM, Pelosi P, Soto L, Tomicic V, D'Empaire G, Matamis D, Abroug F, Moreno RP, Soares MA, Arabi Y, Sandi F, Jibaja M, Amin P, Koh Y, Kuiper MA, Bülow H-H, Zeggwagh AA, Anzueto A (2013) Evolution of mortality over time in patients receiving mechanical ventilation. Am J Respir Crit Care Med 188(2):220–230
2. Mehta A, Syeda SN, Wiener RS, Walkey AJ (2014) Temporal trends in invasive mechanical ventilation: severe sepsis/pneumonia, heart failure and chronic obstructive pulmonary disease. In: B23. Clinical trials and outcomes, vols 271. American Thoracic Society, pp. A2537–A2537
3. Stefan MS, Shieh M-S, Pekow PS, Rothberg MB, Steingrub JS, Lagu T, Lindenauer PK (2013) Epidemiology and outcomes of acute respiratory failure in the United States, 2001 to 2009: a national survey. J Hosp Med 8(2):76–82
4. Traoré O, Liotier J, Souweine B (2005) Prospective study of arterial and central venous catheter colonization and of arterial- and central venous catheter-related bacteremia in intensive care units. Crit Care Med 33(6):1276–1280
5. Gershengorn HB, Garland A, Kramer A, Scales DC, Rubenfeld G, Wunsch H (2014) Variation of arterial and central venous catheter use in United States intensive care units. Anesthesiology 120(3):650–664
6. Gershengorn HB, Wunsch H, Scales DC, Zarychanski R, Rubenfeld G, Garland A (2014) Association between arterial catheter use and hospital mortality in intensive care units. JAMA Intern Med 174(11):1746–1754
7. Maki DG, Kluger DM, Crnich CJ (2006) The risk of bloodstream infection in adults with different intravascular devices: a systematic review of 200 published prospective studies. Mayo Clin Proc 81(9):1159–1171
8. Scheer BV, Perel A, Pfeiffer UJ (2002) Clinical review: complications and risk factors of peripheral arterial catheters used for haemodynamic monitoring in anaesthesia and intensive care medicine. Crit Care 6(3):199–204
9. Concato J, Shah N, Horwitz RI (2000) Randomized, controlled trials, observational studies, and the hierarchy of research designs. N Engl J Med 342(25):1887–1892
10. Ho PM, Peterson PN, Masoudi FA (2008) Evaluating the evidence is there a rigid hierarchy? Circulation 118(16):1675–1684
11. Smith GCS, Pell JP (2003) Parachute use to prevent death and major trauma related to gravitational challenge: systematic review of randomised controlled trials. BMJ 327 (7429):1459–1461
12. Booth CM, Tannock IF (2014) Randomised controlled trials and population-based observational research: partners in the evolution of medical evidence. Br J Cancer 110 (3):551–555

13. Scott DJ, Lee J, Silva I, Park S, Moody GB, Celi LA, Mark RG (2013) Accessing the public MIMIC-II intensive care relational database for clinical research. BMC Med Inform Decis Mak 13(1):9
14. Healthcare Cost and Utilization Project and Agency for Healthcare Research and Quality. Overview of the National (Nationwide) Inpatient Sample (NIS)
15. Doll R, Hill AB (1954) The mortality of doctors in relation to their smoking habits; a preliminary report. Br Med J 1(4877):1451–1455
16. Alberg AJ, Samet JM (2003) Epidemiology of lung cancer. Chest 123(1 Suppl):21S–49S
17. Maldonado G, Greenland S (2002) Estimating causal effects. Int J Epidemiol 31(2):422–429
18. Wilson PWF, D'Agostino RB, Levy D, Belanger AM, Silbershatz H, Kannel WB (1998) Prediction of coronary heart disease using risk factor categories. Circulation 97(18):1837–1847
19. Zimmerman JE, Kramer AA, McNair DS, Malila FM (2006) Acute physiology and chronic health evaluation (APACHE) IV: hospital mortality assessment for today's critically ill patients. Crit Care Med 34(5):1297–1310
20. Le Gall JR, Lemeshow S, Saulnier F (1993) A new simplified acute physiology score (SAPS II) based on a European/North American multicenter study. JAMA 270(24):2957–2963
21. Vincent JL, Moreno R, Takala J, Willatts S, De Mendonça A, Bruining H, Reinhart CK, Suter PM, Thijs LG (1996) The SOFA (sepsis-related organ failure assessment) score to describe organ dysfunction/failure. On behalf of the working group on sepsis-related problems of the European society of intensive care medicine. Intensive Care Med 22(7):707–710
22. Tricoci P, Allen JM, Kramer JM, Califf RM, Smith SC (2009) Scientific evidence underlying the ACC/AHA clinical practice guidelines. JAMA 301(8):831–841
23. Del Fiol G, Workman TE, Gorman PN (2014) Clinical questions raised by clinicians at the point of care: a systematic review. JAMA Intern Med 174(5):710–718

Chapter 10
Defining the Patient Cohort

Ari Moskowitz and Kenneth Chen

Learning Objectives

- Understand the process of cohort selection using large, retrospective databases.
- Learn about additional specific skills in cohort building including data visualization and natural language processing (NLP).

10.1 Introduction

A critical first step in any observational study is the selection of an appropriate patient cohort for analysis. The importance of investing considerable time and effort into selection of the study population cannot be overstated. Failure to identify areas of potential bias, confounding, and missing data up-front can lead to considerable downstream inefficiencies. Further, care must be given to selecting a population of patients tailored to the research question of interest in order to properly leverage the tremendous amount of data captured by Electronic Health Records (EHRs).

In the following chapter we will focus on selection of the study cohort. Specifically, we will review the basics of observational study design with a focus on types of data often encountered in EHRs. Commonly used instrumental variables will be highlighted—they are variables used to control for confounding and measurement error in observational studies. Further, we will discuss how to utilize a combination of data-driven techniques and clinical reasoning in cohort selection. The chapter will conclude with a continuation of the worked example started in part

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-43742-2_10) contains supplementary material, which is available to authorized users.

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_10

one of this section where we will discuss how the cohort of patients was selected for the study of arterial line placement in the intensive care unit [1].

10.2 PART 1—Theoretical Concepts

10.2.1 Exposure and Outcome of Interest

These notions are discussed in detail in Chap. 9—"Formulating the Research Question". Data mining in biomedical research utilizes a retrospective approach wherein the exposure and outcome of interest occur prior to patient selection. It is critically important to tailor the exposure of interest sought to the clinical question at hand. Selecting an overly broad exposure may allow for a large patient cohort, but at the expense of result accuracy. Similarly, being too specific in the choice of exposure may allow for accuracy but at the expense of sample size and generalizability.

The selection of an exposure of interest is the first step in determining the patient cohort. In general, the exposure of interest can be thought of as patient-centric, episode-centric, or encounter centric. This terminology was developed by the data warehousing firm Health Catalyst for their Cohort Builder tool and provides a reasonable framework for identifying an exposure of interest. Patient-centric exposures focus on traits intrinsic to a group of patients. These can include demographic traits (e.g. gender) or medical comorbidities (e.g. diabetes). In contrast, episode-centric exposures are transient conditions requiring a discrete treatment course (e.g. sepsis). Encounter-centric exposures refer to a single intervention (e.g. arterial line placement) [2]. Although encounter-specific exposures tend to be simpler to isolate, the choice of exposure should be determined by the specific hypothesis under investigation.

The outcome of interest should be identified a priori. The outcome should relate naturally to the exposure of interest and be as specific as possible to answer the clinical question at hand. Care must be taken to avoid identifying spurious correlations that have no pathophysiologic underpinnings (see for instance the examples of spurious correlations shown on http://tylervigen.com). The relationship sought must be grounded in biologic plausibility. Broad outcome measures, such as mortality and length-of-stay, may be superficially attractive but ultimately confounded by too many variables. Surrogate outcome measures (e.g. change in blood pressure, duration of mechanical ventilation) can be particularly helpful as they relate more closely to the exposure of interest and are less obscured by confounding.

As EHRs are not frequently oriented towards data mining and analysis, identifying an exposure of interest can be challenging. Structured numerical data, such as laboratory results and vital signs, are easily searchable with standard querying techniques. Leveraging unstructured data such as narrative notes and radiology reports can be more difficult and often requires the use of natural language processing (NLP) tools. In order to select a specific patient phenotype from a large, heterogeneous group of patients, it can be helpful to leverage both structured and unstructured data forms.

Once an exposure of interest is selected, the investigator must consider how to utilize one or a combination of these data types to isolate the desired study cohort for analysis. This can be done using a combination of data driven techniques and clinical reasoning as will be reviewed later in the chapter.

10.2.2 Comparison Group

In addition to isolating patients mapping to the exposure of interest, the investigator must also identify a comparison group. Ideally, this group should be comprised of patients phenotypically similar to those in the study cohort but who lack the exposure of interest. The selected comparison cohort should be at equal risk of developing the study outcome. In observational research, this can be accomplished notably via propensity score development (Chap. 23—"Propensity Score Analysis"). In general, the comparison group ought to be as large as or larger than the study cohort to maximize the power of the study. It is possible to select too many features on which to 'match' the comparison and study cohorts thereby reducing the number of patients available for the comparison cohort. Care must be taken to prevent over-matching.

In select cases, investigators can take advantage of natural experiments in which circumstances external to the EHR readily establish a study cohort and a comparison group. These so called 'instrumental variables' can include practice variations between care units, hospitals, and even geographic regions. Temporal relationships (i.e. before-and-after) relating to quality improvement initiatives or expert guideline releases can also be leveraged as instrumental variables. Investigators should be on the lookout for these highly useful tools.

10.2.3 Building the Study Cohort

Isolating specific patient phenotypes for inclusion in the study and comparison cohorts requires a combination of clinical reasoning and data-driven techniques. A close working relationship between clinicians and data scientists is an essential component of cohort selection using EHR data.

The clinician is on the frontline of medical care and has direct exposure to complex clinical scenarios that exist outside the realm of the available evidence-base. According to a 2011 Institute of Medicine Committee Report, only 10–20 % of clinical decisions are evidence based [3]. Nearly 50 % of clinical practice guidelines rely on expert opinion rather than experimental data [4]. In this 'data desert' it is the role of the clinician to identify novel research questions important for direct clinical care [5]. These questions lend themselves naturally to the isolation of an exposure of interest.

Once a clinical question and exposure of interest have been identified, the clinician and data scientist will need to set about isolating a patient cohort. Phenotype querying of structured and unstructured data can be complex and requires frequent tuning of the search criteria. Often multiple, complementary queries are required in order to isolate the specific group of interest. In addition, the research team must consider patient 'uniqueness' in that some patients have multiple ICU admissions both during a single hospitalization and over repeat hospital visits. If the same patient is included more than once in a study cohort, the assumption of independent measures is lost.

Researchers must pay attention to the necessity to exclude some patients on the grounds of their background medical history or pathological status, such as pregnancy for example. Failing to do so could introduce confounders and corrupt the causal relationship of interest.

In one example from a published MIMIC-II study, the investigators attempted to determine whether proton pump inhibitor (PPI) use was associated with hypomagnesaemia in critically-ill patients in the ICU [6]. The exposure of interest in this study was 'PPI use.' A comparison group of patients who were exposed to an alternative acid-reducing agent (histamine-2 receptor antagonists) and a comparison group not receiving any acid reducing medications were identified. The outcome of interest was a low magnesium level. In order to isolate the study cohort in this case, queries had to be developed to identify:

1. First ICU admission for each patient
2. PPI use as identified through NLP analysis of the 'Medication' section of the admission History and Physical
3. Conditions likely to influence PPI use and/or magnesium levels (e.g. diarrheal illness, end-stage renal disease)
4. Patients who were transferred from other hospitals as medications received at other hospitals could not be accounted for (patients excluded)
5. Patients who did not have a magnesium level within 36-h of ICU admission (patients excluded)
6. Patients missing comorbidity data (patients excluded)
7. Potential confounders including diuretic use

The SQL queries corresponding to this example are provided under the name "SQL_cohort_selection".

Maximizing the efficiency of data querying from EHRs is an area of active research and development. As an example, the Informatics for Integrating Biology and the Bedside (i2b2) network is an NIH funded program based at Partner's Health Center (Boston, MA) that is developing a framework for simplifying data querying and extraction from EHRs. Software tools developed by i2b2 are free to download and promise to simplify the isolation of a clinical phenotype from raw EHR data https://www.i2b2.org/about/index.html. This and similar projects should help simplify the large number of queries necessary to develop a study cohort [7].

10.2.4 Hidden Exposures

Not all exposures of interest can be identified directly from data contained within EHRs. In these circumstances, investigators need to be creative in identifying recorded data points that track closely with the exposure of interest. Clinical reasoning in these circumstances is important.

For instance, a research team using the MIMIC II database selected ‘atrial fibrillation with rapid ventricular response receiving a rate control agent’ as the exposure of interest. Atrial fibrillation is a common tachyarrhythmia in critically-ill populations that has been associated with worse clinical outcomes. Atrial fibrillation with rapid ventricular response is often treated with one of three rate control agents: metoprolol, diltiazem, or amiodarone. Unfortunately, ‘atrial fibrillation with rapid ventricular response’ is not a structured variable in the EHR system connected to the MIMIC II database. Performing an NLP search for the term ‘atrial fibrillation with rapid ventricular response’ in provider notes and discharge summaries is feasible however would not provide the temporal resolution needed with respect to drug administration.

To overcome this obstacle, investigators generated an algorithm to indirectly identify the ‘hidden’ exposure. A query was developed to isolate the first dose of an intravenous rate control agent (metoprolol, diltiazem, or amiodarone) received by a unique patient in the ICU. Next, it was determined whether the heart rate of the patient within one-hour of recorded drug administration was >110 beats per minute. Finally, an NLP algorithm was used to search the clinical chart for mention of atrial fibrillation. Those patients meeting all three conditions were included in the final study cohort. Examples of the Matlab code used to identify the cohort of interest is provided (function “Afib”), as well as Perl code for NLP (function “NLP”).

10.2.5 Data Visualization

Graphic representation of alphanumeric EHR data can be particularly helpful in establishing the study cohort. Data visualization makes EHR data more accessible and allows for the rapid identification of trends otherwise difficult to identify. It also promotes more effective communication both amongst research team members and between the research team and a general audience not accustomed to ‘Big Data’ investigation. These principles are discussed more extensively in Chap. 15 of this textbook “Exploratory Data Analysis”.

In the above mentioned project exploring the use of rate control agents for atrial fibrillation with rapid ventricular response, one outcome of interest was time until control of the rapid ventricular rate. Unfortunately, the existing literature does not provide specific guidance in this area. Using data visualization, a group consensus was reached that rate control would be defined as a heart <110 for at least 90 % of the time over a 4-h period. Although some aspects of this definition are arbitrary, data visualization allowed for all team members to come to an agreement on what definition was the most statistically and clinically defensible.

10.2.6 Study Cohort Fidelity

Query algorithms are generally unable to boast 100 % accuracy for identifying the sought patient phenotype. False positives and false negatives are expected. In order to guarantee the fidelity of the study cohort, manually reviewing a random subset of selected patients can be helpful. Based on the size of the study cohort, 5–10 % of clinical charts should be reviewed to ensure the presence or absence of the exposure of interest. This task should be accomplished by a clinician. If resources permit, two clinician reviewers can be tasked with this role and their independent results compared using a Kappa statistic.

Ultimately, the investigators can use the 'gold standard' of manual review to establish a Receiver Operating Characteristic (ROC). An area-under the ROC curve of >0.80 indicates 'good' accuracy of the algorithm and should be used as an absolute minimum of algorithm fidelity. If the area under the ROC curve is <0.80, a combination of data visualization techniques and clinical reasoning should be used to better tune the query algorithm to the exposure of interest.

10.3 PART 2—Case Study: Cohort Selection

In the case study presented, the authors analyzed the effect of indwelling arterial catheters (IACs) in hemodynamically stable patients with respiratory failure using multivariate data. They identified the encounter-centric 'arterial catheter placement' as their exposure of interest. IACs are used extensively in the intensive care unit for beat-to-beat measuring of blood pressure and are thought to be more accurate and reliable than standard, non-invasive blood pressure monitoring. They also have the added benefit of allowing for simpler arterial blood gas collection which can reduce the need for repeated venous punctures. Given their invasive nature, however, IACs carry risks of bloodstream infection and vascular injury. The primary outcome of interest selected was 28-day mortality with secondary outcomes that included ICU and hospital length-of-stay, duration of mechanical ventilation, and mean number of blood gas measurements made.

The authors elected to focus their study on patients requiring mechanical ventilation that did not require vasopressor and were not admitted for sepsis. In patients requiring mechanical ventilation, the dual role of IACs to allow for beat-to-beat blood pressure monitoring and to simplify arterial blood gas collection is thought to be particularly important. Patients with vasopressor requirements and/or sepsis were excluded as invasive arterial catheters are needed in this population to assist with the rapid titration of vasoactive agents. In addition, it would be difficult to identify enough patients requiring vasopressors or admitted for sepsis, who did not receive an IAC.

The authors began their cohort selection with all 24,581 patients included in the MIMIC II database. For patients with multiple ICU admissions, only the first ICU admission was used to ensure independence of measurements. The function "cohort1" contains the SQL query corresponding to this step. Next, the patients who required mechanical ventilation within the first 24-h of their ICU admission and received mechanical ventilation for at least 24-h stay were isolated (function "cohort2"). After identifying a cohort of patients requiring mechanical ventilation, the authors queried for placement of an IAC sited after initiation of mechanical ventilation (function "cohort3"). As a majority of patients in the cardiac surgery recovery unit had an IAC placed prior to ICU admission, all patients from the cardiac surgical ICU were excluded from the analysis (function "cohort4"). In order to exclude patients admitted to the ICU with sepsis, the authors utilized the Angus criteria (function "cohort5"). Finally, patients requiring vasopressors during their ICU admission were excluded (function "cohort6").

The comparison group of patients who received mechanical ventilation for at least 24-h within the first 24-h of their ICU admission but did not have an IAC placed was identified. Ultimately, there were 984 patients in the group who received an IAC and 792 patients who did not. These groups were compared using propensity matching techniques described in the Chap. 23—"Propensity Score Analysis".

Ultimately, this cohort consists of unique identifiers of patients meeting the inclusion criteria. Other researchers may be interested in accessing this particular cohort in order to replicate the study results or address a different research questions. The MIMIC website will in the future provide the possibility for investigators to share cohorts of patients, thus allowing research teams to interact and build upon other's work.

Take Home Messages

- Take time to characterize the exposure and outcomes of interest pre-hoc
- Utilize both structured and unstructured data to isolate your exposure and outcome of interest. NLP can be particularly helpful in analyzing unstructured data
- Data visualization can be very helpful in facilitating communication amongst team members

References

1. Hsu DJ, Feng M, Kothari R, Zhou H, Chen KP, Celi LA (2015) The association between indwelling arterial catheters and mortality in hemodynamically stable patients with respiratory failure: a propensity score analysis. Chest 148(6):1470–1476
2. Merkley K (2013) Defining patient populations using analytical tools: cohort builder and risk stratification. Health Catalyst, 21 Aug 2013
3. Institute of Medicine (US) Committee on Standards for Developing Trustworthy Clinical Practice Guidelines (2011) Clinical practice guidelines we can trust. National Academies Press (US), Washington (DC)
4. Committee on the Learning Health Care System in America and Institute of Medicine (2013) Best care at lower cost: the path to continuously learning health care in America. National Academies Press (US), Washington (DC)
5. Moskowitz A, McSparron J, Stone DJ, Celi LA (2015) Preparing a new generation of clinicians for the era of big data. Harv Med Stud Rev 2(1):24–27
6. Danziger J, William JH, Scott DJ, Lee J, Lehman L, Mark RG, Howell MD, Celi LA, Mukamal KJ (2013) Proton-pump inhibitor use is associated with low serum magnesium concentrations. Kidney Int 83(4):692–699
7. Jensen PB, Jensen LJ, Brunak S (2012) Mining electronic health records: towards better research applications and clinical care. Nat Rev Genet 13(6):395–405

Chapter 11
Data Preparation

Tom Pollard, Franck Dernoncourt, Samuel Finlayson and Adrian Velasquez

Learning Objectives

- Become familiar with common categories of medical data.
- Appreciate the importance of collaboration between caregivers and data analysts.
- Learn common terminology associated with relational databases and plain text data files.
- Understand the key concepts of reproducible research.
- Get practical experience in querying a medical database.

11.1 Introduction

Data is at the core of all research, so robust data management practices are important if studies are to be carried out efficiently and reliably. The same can be said for the management of the software used to process and analyze data. Ensuring good practices are in place at the beginning of a study is likely to result in significant savings further down the line in terms of time and effort [1, 2].

While there are well-recognized benefits in tools and practices such as version control, testing frameworks, and reproducible workflows, there is still a way to go before these become widely adopted in the academic community. In this chapter we discuss some key issues to consider when working with medical data and highlight some approaches that can make studies collaborative and reproducible.

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_11

11.2 Part 1—Theoretical Concepts

11.2.1 Categories of Hospital Data

Data is routinely collected from several different sources within hospitals, and is generally optimized to support clinical activities and billing rather than research. Categories of data commonly found in practice are summarized in Table 11.1 and discussed below:

- Billing data generally consists of the codes that hospitals and caregivers use to file claims with their insurance providers. The two most common coding systems are the International Statistical Classification of Diseases and Related

Table 11.1 Overview of common categories of hospital data and common issues to consider during analysis

Category	Examples	Common issues to consider
Demographics	Age, gender, ethnicity, height, weight	Highly sensitive data requiring careful de-identification. Data quality in fields such as ethnicity may be poor
Laboratory	Creatinine, lactate, white blood cell count, microbiology results	Often no measure of sample quality. Methods and reagents used in tests may vary between units and across time
Radiographic images and associated reports	X-rays, computed tomography (CT) scans, echocardiograms	Protected health information, such as names, may be written on slides. Templates used to generate reports may influence content
Physiologic data	Vital signs, electrocardiography (ECG) waveforms, electroencephalography (EEG) waveforms	Data may be pre-processed by proprietary algorithms. Labels may be inaccurate (for example, "fingerstick glucose" measurements may be made with venous blood)
Medication	Prescriptions, dose, timing	May list medications that were ordered but not given. Time stamps may describe point of order not administration
Diagnosis and procedural codes	International Classification of Diseases (ICD) codes, Diagnosis Related Groups (DRG) codes, Current Procedural Terminology (CPT) codes	Often based on a retrospective review of notes and not intended to indicate a patient's medical status. Subject to coder biases. Limited by suitability of codes
Caregiver and procedural notes	Admission notes, daily progress notes, discharge summaries, Operative reports	Typographical errors. Context is important (for example, diseases may appear in discussion of family history). Abbreviations and acronyms are common

Health Problems, commonly abbreviated the International Classification of Disease (ICD), which is maintained by the World Health Organization, and the Current Procedural Terminology (CPT) codes maintained by the American Medical Association. These hierarchical terminologies were designed to provide standardization for medical classification and reporting.

- Charted physiologic data, including information such as heart rate, blood pressure, and respiratory rate collected at the bedside. The frequency and breadth of monitoring is generally related to the level of care. Data is often archived at a lower rate than it is sampled (for example, every 5–10 min) using averaging algorithms which are frequently proprietary and undisclosed.
- Notes and reports, created to record patient progress, summaries a patient stay upon discharge, and provide findings from imaging studies such as x-rays and echocardiograms. While the fields are "free text", notes are often created with the help of a templating system, meaning they may be partially structured.
- Images, such as those from x-rays, computerized axial tomography (CAT/CT) scans, echocardiograms, and magnetic resonance imaging.
- Medication and laboratory data. Orders for drugs and laboratory studies are entered by the caregiver into a physician order entry system, which are then fulfilled by laboratory or nursing staff. Depending on the system, some timestamps may refer to when the physician placed the order and others may refer to when the drug was administered or the lab results were reported. Some drugs may be administered days or weeks after first prescribed while some may not be administered at all.

11.2.2 Context and Collaboration

One of the greatest challenges of working with medical data is gaining knowledge of the context in which data is collected. For this reason we cannot emphasize enough the importance of collaboration between both hospital staff and research analysts. Some examples of common issues to consider when working with medical data are outlined in Table 11.1 and discussed below:

- Billing codes are not intended to document a patient's medical status or treatment from a clinical perspective and so may not be reliable [3]. Coding practices may be influenced by issues such as financial compensation and associated paperwork, deliberately or otherwise.
- Timestamps may differ in meaning for different categories of data. For example, a timestamp may refer to the point when a measurement was made, when the measurement was entered into the system, when a sample was taken, or when results were returned by a laboratory.
- Abbreviations and misspelled words appear frequently in free text fields. The string "pad", for example, may refer to either "peripheral artery disease" or to an

absorptive bed pad, or even a diaper pad. In addition, notes frequently mention diseases that are found in the patient's family history, but not necessarily the patient, so care must be taken when using simple text searches.
- Labels that describe concepts may not be accurate. For example, during preliminary investigations for an unpublished study to assess accuracy of fingertip glucose testing, it was discovered that caregivers would regularly take "fingerstick glucose" measurements using vascular blood where it was easily accessible, to avoid pricking the finger of a patient.

Each hospital brings its own biases to the data too. These biases may be tied to factors such as the patient populations served, the local practices of caregivers, or to the type of services provided. For example:

- Academic centers often see more complicated patients, and some hospitals may tend to serve patients of a specific ethnic background or socioeconomic status.
- Follow up visits may be less common at referral centers and so they may be less likely to detect long-term complications.
- Research centers may be more likely to place patients on experimental drugs not generally used in practice.

11.2.3 Quantitative and Qualitative Data

Data is often described as being either quantitative or qualitative. Quantitative data is data that can be measured, written down with numbers and manipulated numerically. Quantitative data can be discrete, taking only certain values (for example, the integers 1, 2, 3), or continuous, taking any value (for example, 1.23, 2.59). The number of times a patient is admitted to a hospital is discrete (a patient cannot be admitted 0.7 times), while a patient's weight is a continuous (a patient's weight could take any value within a range).

Qualitative data is information which cannot be expressed as a number and is often used interchangeably with the term "categorical" data. When there is not a natural ordering of the categories (for example, a patient's ethnicity), the data is called nominal. When the categories can be ordered, these are called ordinal variables (for example, severity of pain on a scale). Each of the possible values of a categorical variable is commonly referred to as a level.

11.2.4 Data Files and Databases

Data is typically made available through a database or as a file which may have been exported from a database. While there are many different kinds of databases and data files in use, relational databases and comma separated value (CSV) files are perhaps the most common.

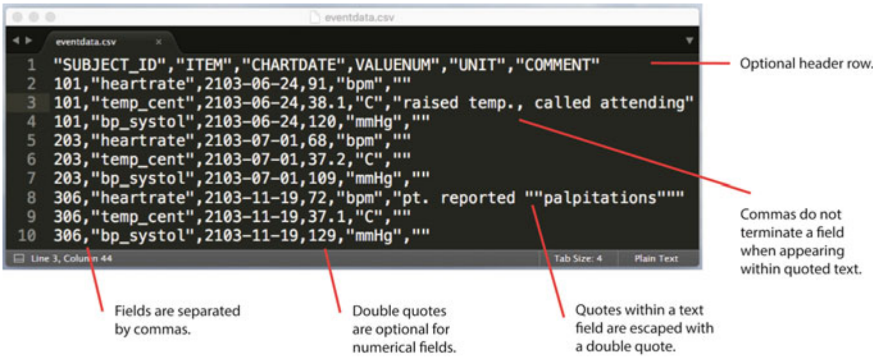
Comma Separated Value (CSV) Files

Comma separated value (CSV) files are a plain text format used for storing data in a tabular, spreadsheet-style structure. While there is no hard and fast rule for structuring tabular data, it is usually considered good practice to include a header row, to list each variable in a separate column, and to list observations in rows [4].

As there is no official standard for the CSV format, the term is used somewhat loosely, which can often cause issues when seeking to load the data into a data analysis package. A general recommendation is to follow the definition for CSVs set out by the Internet Engineering Task Force in the RFC 4180 specification document [5]. Summarized briefly, RFC 4180 specifies that:

- files may optionally begin with a header row, with each field separated by a comma;
- Records should be listed in subsequent rows. Fields should be separated by commas, and each row should be terminated with a line break;
- fields that contain numbers may be optionally enclosed within double quotes;
- fields that contain text (“strings”) should be enclosed within double quotes;
- If a double quote appears inside a string of text then it must be escaped with a preceding double quote.

The CSV format is popular largely because of its simplicity and versatility. CSV files can be edited with a text editor, loaded as a spreadsheet in packages such as Microsoft Excel, and imported and processed by most data analysis packages. Often CSV files are an intermediate data format used to hold data that has been extracted from a relational database in preparation for analysis. Figure 11.1 shows an annotated example of a CSV file formatted to the RFC 4180 specification.



```
"SUBJECT_ID","ITEM","CHARTDATE",VALUENUM","UNIT","COMMENT"
101,"heartrate",2103-06-24,91,"bpm",""
101,"temp_cent",2103-06-24,38.1,"C","raised temp., called attending"
101,"bp_systol",2103-06-24,120,"mmHg",""
203,"heartrate",2103-07-01,68,"bpm",""
203,"temp_cent",2103-07-01,37.2,"C",""
203,"bp_systol",2103-07-01,109,"mmHg",""
306,"heartrate",2103-11-19,72,"bpm","pt. reported ""palpitations"""
306,"temp_cent",2103-11-19,37.1,"C",""
306,"bp_systol",2103-11-19,129,"mmHg",""
```

Optional header row.

Commas do not terminate a field when appearing within quoted text.

Fields are separated by commas.

Double quotes are optional for numerical fields.

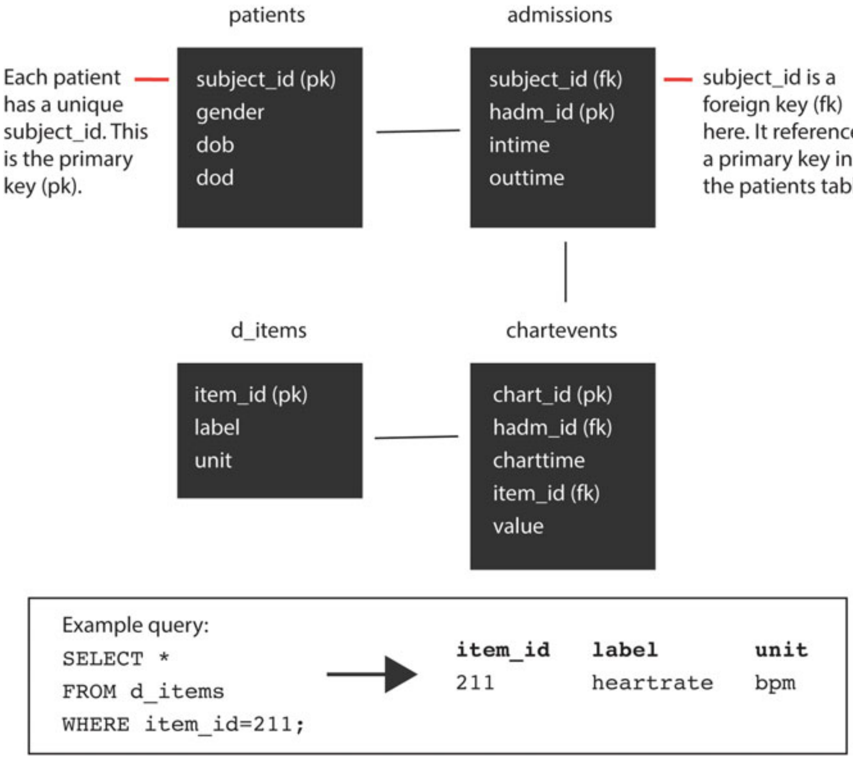
Quotes within a text field are escaped with a double quote.

Fig. 11.1 Comma separated value (CSV) file formatted to the RFC 4180 specification

Relational Databases

There are several styles of database in use today, but probably the most widely implemented is the "relational database". Relational databases can be thought of as a collection of tables which are linked together by shared keys. Organizing data across tables can help to maintain data integrity and enable faster analysis and more efficient storage.

The model that defines the structure and relationships of the tables is known as a "database schema". Giving a simple example of a hospital database with four tables, it might comprise of: Table 1, a list of all patients; Table 2, a log of hospital admissions; Table 3, a list of vital sign measurements; Table 4, a dictionary of vital sign codes and associated labels. Figure 11.2 demonstrates how these tables can be linked with primary and foreign keys. Briefly, a primary key is a unique identifier within a table. For example, subject_id is the primary key in the patients table,



```
SELECT *
FROM d_items
WHERE item_id=211;
```

```
item_id   label       unit
211       heartrate   bpm
```

Fig. 11.2 Relational databases consist of multiple data tables linked by primary and foreign keys. The patients table lists unique patients. The admissions table lists unique hospital admissions. The chartevents table lists charted events such as heart rate measurements. The d_items table is a dictionary that lists item_ids and associated labels, as shown in the example query. *pk* is primary key. *fk* is foreign key

because each patient is listed only once. A foreign key in one table points to a primary key in another table. For example, subject_id in the admissions table is a foreign key, because it references the primary key in the patients table.

Extracting data from a database is known as "querying" the database. The programming language commonly used to create a query is known as "Structured Query Language" or SQL. While the syntax of SQL is straightforward, queries are at times challenging to construct as a result of the conceptual reasoning required to join data across multiple tables.

There are many different relational database systems in regular use. Some of these systems such as Oracle Database and Microsoft SQL Server are proprietary and may have licensing costs. Other systems such as PostgreSQL and MySQL are open source and free to install. The general principle behind the databases is the same, but it is helpful to be aware that programming syntax varies slightly between systems.

11.2.5 Reproducibility

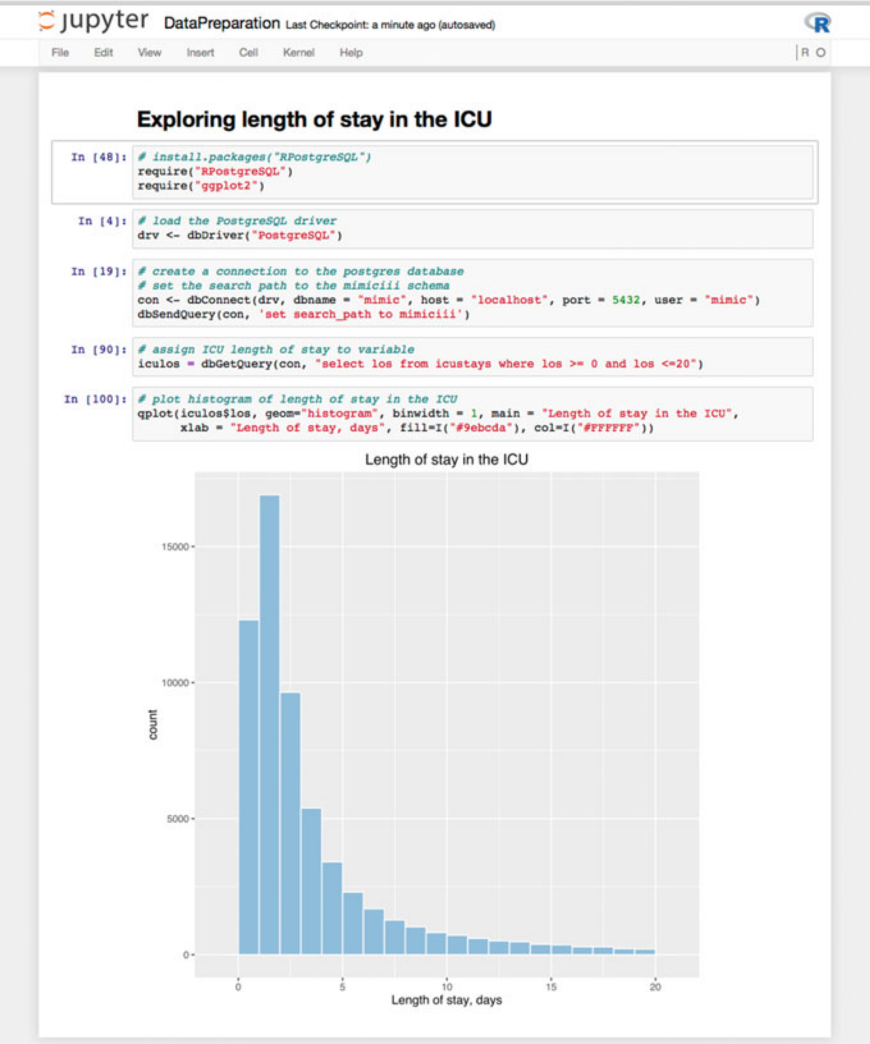
Alongside a publishing system that emphasizes interpretation of results over detailed methodology, researchers are under pressure to deliver regular "high-impact" papers in order to sustain their careers. This environment may be a contributor to the widely reported "reproducibility crisis" in science today [6, 7].

Our response should be to ensure that studies are, as far as possible, reproducible. By making data and code accessible, we can more easily detect and fix inevitable errors, help each other to learn from our methods, and promote better quality research.

When practicing reproducible research, the source data should not be modified. Editing the raw data destroys the chain of reproducibility. Instead, code is used to process the data so that all of the steps that take an analysis from source to outcome can be reproduced.

Code and data should be well documented and the terms of reuse should be made clear. It is typical to provide a plain text "README" file that gives an introduction to the analysis package, along with a "LICENSE" file describing the terms of reuse. Tools such as Jupyter Notebook, Sweave, and Knitr can be used to interweave code and text to produce clearly documented, reproducible studies, and are becoming increasingly popular in the research community (Fig. 11.3).

Version control systems such as Git can be used to track the changes made to code over time and are also becoming an increasingly popular tool for researchers [8]. When working with a version control system, a commit log provides a record of changes to code by contributor, providing transparency in the development process and acting as a useful tool for uncovering and fixing bugs.



Exploring length of stay in the ICU

```
In [48]: # install.packages("RPostgreSQL")
         require("RPostgreSQL")
         require("ggplot2")

In [4]:  # load the PostgreSQL driver
         drv <- dbDriver("PostgreSQL")

In [19]: # create a connection to the postgres database
         # set the search path to the mimiciii schema
         con <- dbConnect(drv, dbname = "mimic", host = "localhost", port = 5432, user = "mimic")
         dbSendQuery(con, 'set search_path to mimiciii')

In [90]: # assign ICU length of stay to variable
         iculos = dbGetQuery(con, "select los from icustays where los >= 0 and los <=20")

In [100]: # plot histogram of length of stay in the ICU
         qplot(iculos$los, geom="histogram", binwidth = 1, main = "Length of stay in the ICU",
               xlab = "Length of stay, days", fill=I("#9ebcda"), col=I("#FFFFFF"))
```

Fig. 11.3 Jupyter Notebooks enable documentation and code to be combined into a reproducible analysis. In this example, the length of ICU stay is loaded from the MIMIC-III (v1.3) database and plotted as a histogram [11]

Collaboration is also facilitated by version control systems. Git provides powerful functionality that facilitates distribution of code and allows multiple people to work together in synchrony. Integration with Git hosting services such as Github provide a simple mechanism for backing up content, helping to reduce the risk of data loss, and also provide tools for tracking issues and tasks [8, 9].

11.3 Part 2—Practical Examples of Data Preparation

11.3.1 MIMIC Tables

In order to carry out the study on the effect of indwelling arterial catheters as described in the previous chapter, we use the following tables in the MIMIC-III clinical database:

- The chartevents table, the largest table in the database. It contains all data charted by the bedside critical care system, including physiological measurements such as heart rate and blood pressure, as well as the settings used by the indwelling arterial catheters.
- The patients table, which contains the demographic details of each patient admitted to an intensive care unit, such as gender, date of birth, and date of death.
- The icustays table, which contains administrative details relating to stays in the ICU, such as the admission time, discharge time, and type of care unit.

Before continuing with the following exercises, we recommend familiarizing yourself with the MIMIC documentation and in particular the table descriptions, which are available on the MIMIC website [10].

11.3.2 SQL Basics

An SQL query has the following format:

```
SELECT [columns]
FROM [table_name]
WHERE [conditions];
```

The result returned by the query is a list of rows. The following query lists the unique patient identifiers (subject_ids) of all female patients:

```
SELECT subject_id
FROM patients
WHERE gender = 'F';

-- returns:
 subject_id
------------
        654
        655
        656
        ...
```

We often need to specify more than one condition. For instance, the following query lists the subject_ids whose first or last care unit was a coronary care unit (CCU):

```
SELECT subject_id
FROM icustays
WHERE first_careunit = 'CCU' OR last_careunit = 'CCU';

-- returns:
 subject_id
------------
        109
        109
        111
        ...
```

Since a patient may have been in several ICUs, the same patient ID sometimes appears several times in the result of the previous query. To return only distinct rows, use the DISTINCT keyword:

```
SELECT DISTINCT subject_id
FROM icustays
WHERE first_careunit = 'CCU' OR last_careunit = 'CCU';

-- returns:
 subject_id
------------
      25949
       6158
      27223
      ...
```

To count how many patients there are in the icustays table, combine DISTINCT with the COUNT keyword. As you can see, if there is no condition, we simply don't use the keyword WHERE:

```
SELECT COUNT(DISTINCT subject_id)
FROM icustays;

-- returns:
 count
-------
 46476
```

Taking a similar approach, we can count how many patients went through the CCU using the query:

```
SELECT COUNT(DISTINCT subject_id)
FROM icustays
WHERE first_careunit = 'CCU' OR last_careunit = 'CCU';

-- returns:
 count
-------
  7314
```

The operator * is used to display all columns. The following query displays the entire icustays table:

```
SELECT *
FROM icustays;

-- returns
subject_id | hadm_id | icustay_id | ...
       109 |  139061 |     257358 | ...
       109 |  172335 |     262652 | ...
       109 |  126055 |     236124 | ...
...
```

The results can be sorted based on one or several columns with ORDER BY. To add a comment in a SQL query, use:

```
SELECT subject_id, hadm_id, icustay_id
FROM icustays
ORDER BY subject_id ASC; -- ASC sorts by ascending number

-- returns:
 subject_id | hadm_id | icustay_id
------------+---------+------------
          2 |  163353 |     243653
          3 |  145834 |     211552
          4 |  185777 |     294638
...
```

11.3.3 Joins

Often we need information coming from multiple tables. This can be achieved using SQL joins. There are several types of join, including INNER JOIN, OUTER JOIN, LEFT JOIN, and RIGHT JOIN. It is important to understand the difference between these joins because their usage can significantly impact query results. Detailed guidance on joins is widely available on the web, so we will not go into further details here. We will however provide an example of an INNER JOIN which selects all rows where the joined key appears in both tables.

Using the INNER JOIN keyword, let's count how many adult patients went through the coronary care unit. To know whether a patient is an adult, we need to use the dob (date of birth) attribute from the patients table. We can use the INNER JOIN to indicate that two or more tables should be combined based on a common attribute, which in our case is subject_id:

```
-- INNER JOIN will only return rows where subject_id
-- appears in the patients table and the icustays table
SELECT p.subject_id
FROM patients p
INNER JOIN icustays i
ON p.subject_id = i.subject_id
WHERE (i.first_careunit = 'CCU' OR i.last_careunit = 'CCU')
   AND (i.intime - p.dob) >= INTERVAL '18' year
ORDER BY subject_id ASC;

-- returns:
 subject_id
------------
         13
         18
         21
         ...
```

Note that:

- we assign an alias to a table to avoid writing its full name throughout the query. In our 0 given the alias 'p'.
- in the SELECT clause, we wrote p.subject_id instead of simply subject_id since both the patients and icustays tables contain the attribute subject_id. If we don't specify from which table subject_id comes from, we would get a "column ambiguously defined" error.
- to identify whether a patient is an adult, we look for differences between intime and dob of 18 years or greater using the INTERVAL keyword.

11.3.4 Ranking Across Rows Using a Window Function

We now focus on the case study. One of the first steps is identifying the first ICU admission for each patient. To do so, we can use the RANK () function to order rows sequentially by intime. Using the PARTITION BY expression allows us to perform the ranking across subject_id windows:

```
SELECT subject_id, icustay_id, intime,
    RANK() OVER (PARTITION BY subject_id ORDER BY intime asc)
FROM icustays;

-- returns:
 subject_id | icustay_id |       intime        | rank
------------+------------+---------------------+------
          6 |     228232 | 2175-05-30 21:30:54 |    1
          7 |     278444 | 2121-05-23 15:35:29 |    1
          7 |     236754 | 2121-05-25 03:26:01 |    2
        ...
```

11.3.5 Making Queries More Manageable Using WITH

To keep SQL queries reasonably short and simple, we can use the WITH keyword. WITH allows us to break a large query into smaller, more manageable chunks. The following query creates a temporary table called “rankedstays” that lists the order of stays for each patient. We then select only the rows in this table where the rank is equal to one (i.e. the first stay) and the patient is aged 18 years or greater:

```
WITH rankedstays AS (
    SELECT subject_id, icustay_id, intime,
        RANK() OVER (PARTITION BY subject_id ORDER BY intime asc)
    FROM icustays
)
SELECT r.subject_id, r.icustay_id, r.intime, r.rank
FROM rankedstays r
INNER JOIN patients p
ON r.subject_id = p.subject_id
WHERE r.rank = 1
AND (r.intime - p.dob) >= INTERVAL '18' year;

-- returns:
 subject_id | icustay_id |       intime        | rank
------------+------------+---------------------+------
          3 |     211552 | 2101-10-20 19:10:11 |    1
          4 |     294638 | 2191-03-16 00:29:31 |    1
          6 |     228232 | 2175-05-30 21:30:54 |    1
        ...
```

References

1. Wilson G, Aruliah DA, Brown CT, Chue Hong NP, Davis M, Guy RT et al (2014) Best practices for scientific computing. PLoS Biol 12(1):e1001745. doi:10.1371/journal.pbio.1001745. http://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1001745
2. Editorial (2012) Must try harder. Nature 483(509). doi:10.1038/483509a. http://www.nature.com/nature/journal/v483/n7391/full/483509a.html
3. Misset B, Nakache D, Vesin A, Darmon M, Garrouste-Orgeas M, Mourvillier B et al (2008) Reliability of diagnostic coding in intensive care patients. Crit Care 12(4):R95. http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2575581/
4. Wickham H (2014) Tidy data. J Stat Softw 59(10):1–23. doi:10.18637/jss.v059.i10. https://www.jstatsoft.org/article/view/v059i10
5. Sustainability of Digital Formats Planning for Library of Congress Collections. Accessed: 24 Feb 2016. CSV, Comma Separated Values (RFC 4180). http://www.digitalpreservation.gov/formats/fdd/fdd000323.shtml
6. Editorial (2013) Unreliable research: trouble at the Lab. Economist. http://www.economist.com/news/briefing/21588057-scientists-think-science-self-correcting-alarming-degree-it-not-trouble
7. Goodman A, Pepe A, Blocker AW, Borgman CL, Cranmer K, Crosas M, et al (2014) Ten simple rules for the care and feeding of scientific data. PLoS Comput Biol 10(4):e1003542. doi:10.1371/journal.pcbi.1003542. http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003542
8. Karthik R (2013) Git can facilitate greater reproducibility and increased transparency in science. Source Code Biol Med 28; 8(1):7. doi:10.1186/1751-0473-8-7. http://scfbm.biomedcentral.com/articles/10.1186/1751-0473-8-7
9. GitHub. https://github.com. Accessed 24 Feb 2016
10. MIMIC website. http://mimic.physionet.org. Accessed 24 Feb 2016
11. MIMIC Code Repository. https://github.com/MIT-LCP/mimic-code. Accessed 24 Feb 2016

Chapter 12
Data Pre-processing

Brian Malley, Daniele Ramazzotti and Joy Tzung-yu Wu

Learning Objectives

- Understand the requirements for a "clean" database that is "tidy" and ready for use in statistical analysis.
- Understand the steps of cleaning raw data, integrating data, reducing and reshaping data.
- Be able to apply basic techniques for dealing with common problems with raw data including missing data inconsistent data, and data from multiple sources.

12.1 Introduction

Data pre-processing consists of a series of steps to transform raw data derived from data extraction (see Chap. 11) into a "clean" and "tidy" dataset prior to statistical analysis. Research using electronic health records (EHR) often involves the secondary analysis of health records that were collected for clinical and billing (non-study) purposes and placed in a study database via automated processes. Therefore, these databases can have many quality control issues. Pre-processing aims at assessing and improving the quality of data to allow for reliable statistical analysis.

Several distinct steps are involved in pre-processing data. Here are the general steps taken to pre-process data [1]:

- Data "cleaning"—This step deals with missing data, noise, outliers, and duplicate or incorrect records while minimizing introduction of bias into the database. These methods are explored in detail in Chaps. 13 and 14.
- "Data integration"—Extracted raw data can come from heterogeneous sources or be in separate datasets. This step reorganizes the various raw datasets into a single dataset that contain all the information required for the desired statistical analyses.

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_12

- "Data transformation"—This step translates and/or scales variables stored in a variety of formats or units in the raw data into formats or units that are more useful for the statistical methods that the researcher wants to use.
- "Data reduction"—After the dataset has been integrated and transformed, this step removes redundant records and variables, as well as reorganizes the data in an efficient and "tidy" manner for analysis.

Pre-processing is sometimes iterative and may involve repeating this series of steps until the data are satisfactorily organized for the purpose of statistical analysis. During pre-processing, one needs to take care not to accidentally introduce bias by modifying the dataset in ways that will impact the outcome of statistical analyses. Similarly, we must avoid reaching statistically significant results through "trial and error" analyses on differently pre-processed versions of a dataset.

12.2 Part 1—Theoretical Concepts

12.2.1 Data Cleaning

Real world data are usually "messy" in the sense that they can be incomplete (e.g. missing data), they can be noisy (e.g. random error or outlier values that deviate from the expected baseline), and they can be inconsistent (e.g. patient age 21 and admission service is neonatal intensive care unit).

The reasons for this are multiple. Missing data can be due to random technical issues with biomonitors, reliance on human data entry, or because some clinical variables are not consistently collected since EHR data were collected for non-study purposes. Similarly, noisy data can be due to faults or technological limitations of instruments during data gathering (e.g. dampening of blood pressure values measured through an arterial line), or because of human error in entry. All the above can also lead to inconsistencies in the data. Bottom line, all of these reasons create the need for meticulous data cleaning steps prior to analysis.

Missing Data

A more detailed discussion regarding missing data will be presented in Chap. 13. Here, we describe three possible ways to deal with missing data [1]:

- Ignore the record. This method is not very effective, unless the record (observation/row) contains several variables with missing values. This approach is especially problematic when the percentage of missing values per variable varies considerably or when there is a pattern of missing data related to an unrecognized underlying cause such as patient condition on admission.

- Determine and fill in the missing value manually. In general, this approach is the most accurate but it is also time-consuming and often is not feasible in a large dataset with many missing values.
- Use an expected value. The missing values can be filled in with predicted values (e.g. using the mean of the available data or some prediction method). It must be underlined that this approach may introduce bias in the data, as the inserted values may be wrong. This method is also useful for comparing and checking the validity of results obtained by ignoring missing records.

Noisy Data

We term noise a random error or variance in an observed variable—a common problem for secondary analyses of EHR data. For example, it is not uncommon for hospitalized patients to have a vital sign or laboratory value far outside of normal parameters due to inadequate (hemolyzed) blood samples, or monitoring leads disconnected by patient movement. Clinicians are often aware of the source of error and can repeat the measurement then ignore the known incorrect outlier value when planning care. However, clinicians cannot remove the erroneous measurement from the medical record in many cases, so it will be captured in the database. A detailed discussion on how to deal with noisy data and outliers is provided in Chap. 14; for now we limit the discussion to some basic guidelines [1].

- Binning methods. Binning methods smooth a sorted data value by considering their ‘neighborhood’, or values around it. These kinds of approaches to reduce noise, which only consider the neighborhood values, are said to be performing local smoothing.
- Clustering. Outliers may be detected by clustering, that is by grouping a set of values in such a way that the ones in the same group (i.e., in the same cluster) are more similar to each other than to those in other groups.
- Machine learning. Data can be smoothed by means of various machine learning approaches. One of the classical methods is the regression analysis, where data are fitted to a specified (often linear) function.

Same as for missing data, human supervision during the process of noise smoothing or outliers detection can be effective but also time-consuming.

Inconsistent Data

There may be inconsistencies or duplications in the data. Some of them may be corrected manually using external references. This is the case, for instance, of errors made at data entry. Knowledge engineering tools may also be used to detect the violation of known data constraints. For example, known functional dependencies among attributes can be used to find values contradicting the functional constraints.

Inconsistencies in EHR result from information being entered into the database by thousands of individual clinicians and hospital staff members, as well as captured from a variety of automated interfaces between the EHR and everything from telemetry monitors to the hospital laboratory. The same information is often entered in different formats by these different sources.

Take, for example, the intravenous administration of 1 g of the antibiotic vancomycin contained in 250 mL of dextrose solution. This single event may be captured in the dataset in several different ways. For one patient this event may be captured from the medication order as the code number (ITEMID in MIMIC) from the formulary for the antibiotic vancomycin with a separate column capturing the dose stored as a numerical variable. However, on another patient the same event could be found in the fluid intake and output records under the code for the IV dextrose solution with an associated free text entered by the provider. This text would be captured in the EHR as, for example "vancomycin 1 g in 250 ml", saved as a text variable (string, array of characters, etc.) with the possibility of spelling errors or use of nonstandard abbreviations. Clinically these are the exact same event, but in the EHR and hence in the raw data, they are represented differently. This can lead to the same single clinical event not being captured in the study dataset, being captured incorrectly as a different event, or being captured multiple times for a single occurrence.

In order to produce an accurate dataset for analysis, the goal is for each patient to have the same event represented in the same manner for analysis. As such, dealing with inconsistency perfectly would usually have to happen at the data entry or data extraction level. However, as data extraction is imperfect, pre-processing becomes important. Often, correcting for these inconsistencies involves some understanding of how the data of interest would have been captured in the clinical setting and where the data would be stored in the EHR database.

12.2.2 Data Integration

Data integration is the process of combining data derived from various data sources (such as databases, flat files, etc.) into a consistent dataset. There are a number of issues to consider during data integration related mostly to possible different standards among data sources. For example, certain variables can be referred by means of different IDs in two or more sources.

In the MIMIC database this mainly becomes an issue when some information is entered into the EHR during a different phase in the patient's care pathway, such as before admission in the emergency department, or from outside records. For example, a patient may have laboratory values taken in the ER before they are admitted to the ICU. In order to have a complete dataset it will be necessary to integrate the patient's full set of lab values (including those not associated with the same MIMIC ICUSTAY identifier) with the record of that ICU admission without repeating or missing records. Using shared values between datasets (such as a hospital stay identifier or a timestamp in this example) can allow for this to be done accurately.

Once data cleaning and data integration are completed, we obtain one dataset where entries are reliable.

12.2.3 Data Transformation

There are many possible transformations one might wish to do to raw data values depending on the requirement of the specific statistical analysis planned for a study. The aim is to transform the data values into a format, scale or unit that is more suitable for analysis (e.g. log transform for linear regression modeling). Here are few common possible options:

Normalization
This generally means data for a numerical variable are scaled in order to range between a specified set of values, such as 0–1. For example, scaling each patient's severity of illness score to between 0 and 1 using the known range of that score in order to compare between patients in a multiple regression analysis.

Aggregation
Two or more values of the same attribute are aggregated into one value. A common example is the transformation of categorical variables where multiple categories can be aggregated into one. One example in MIMIC is to define all surgical patients by assigning a new binary variable to all patients with an ICU service noted to be "SICU" (surgical ICU) or "CSRU" (cardiac surgery ICU).

Generalization
Similar to aggregation, in this case low level attributes are transformed into higher level ones. For example, in the analysis of chronic kidney disease (CKD) patients, instead of using a continuous numerical variable like the patient's creatinine levels, one could use a variable for CKD stages as defined by accepted guidelines.

12.2.4 Data Reduction

Complex analysis on large datasets may take a very long time or even be infeasible. The final step of data pre-processing is data reduction, i.e., the process of reducing the input data by means of a more effective representation of the dataset without compromising the integrity of the original data. The objective of this step is to provide a version of the dataset on which the subsequent statistical analysis will be more effective. Data reduction may or may not be lossless. That is the end database may contain all the information of the original database in more efficient format (such as removing redundant records) or it may be that data integrity is maintained but some information is lost when data is transformed and then only represented in the new form (such as multiple values being represented as an average value).

One common MIMIC database example is collapsing the ICD9 codes into broad clinical categories or variables of interest and assigning patients to them. This reduces the dataset from having multiple entries of ICD9 codes, in text format, for a given patient, to having a single entry of a binary variable for an area of interest to the study, such as history of coronary artery disease. Another example would be in the case of using blood pressure as a variable in analysis. An ICU patient will generally have their systolic and diastolic blood pressure monitored continuously via an arterial line or recorded multiple times per hour by an automated blood pressure cuff. This results in hundreds of data points for each of possibly thousands of study patients. Depending on the study aims, it may be necessary to calculate a new variable such as average mean arterial pressure during the first day of ICU admission.

Lastly, as part of more effective organization of datasets, one would also aim to reshape the columns and rows of a dataset so that it conforms with the following 3 rules of a “tidy” dataset [2, 3]:

1. Each variable forms a column
2. Each observation forms a row
3. Each value has its own cell

“Tidy” datasets have the advantage of being more easily visualized and manipulated for later statistical analysis. Datasets exported from MIMIC usually are fairly “tidy” already; therefore, rule 2 is hardly ever broken. However, sometimes there may still be several categorical values within a column even for MIMIC datasets, which breaks rule 1. For example, multiple categories of marital status or ethnicity under the same column. For some analyses, it is useful to split each categorical values of a variable into their own columns. Fortunately though, we do not often have to worry about breaking rule 3 for MIMIC data as there are not often multiple values in a cell. These concepts will become clearer after the MIMIC examples in Sect. 12.3

12.3 PART 2—Examples of Data Pre-processing in R

There are many tools for doing data pre-processing available, such as R, STATA, SAS, and Python; each differs in the level of programming background required. R is a free tool that is supported by a range of statistical and data manipulation packages. In this section of the chapter, we will go through some examples demonstrating various steps of data pre-processing in R, using data from various MIMIC dataset (SQL extraction codes included). Due to the significant content involved with the data cleaning step of pre-processing, this step will be separately addressed in Chaps. 13 and 14. The examples in this section will deal with some R basics as well as data integration, transformation, and reduction.

12.3.1 R—The Basics

The most common data output from a MIMIC database query is in the form of 'comma separated values' files, with filenames ending in '.csv'. This output file format can be selected when exporting the SQL query results from MIMIC database. Besides '.csv' files, R is also able to read in other file formats, such as Excel, SAS, etc., but we will not go into the detail here.

Understanding 'Data Types' in R

For many who have used other data analysis software or who have a programming background, you will be familiar with the concept of 'data types'.

R strictly stores data in several different data types, called 'classes':

- Numeric – e.g. 3.1415, 1.618
- Integer – e.g. -1, 0, 1, 2, 3
- Character – e.g. "vancomycin", "metronidazole"
- Logical – TRUE, FALSE
- Factors/categorical – e.g. male or female under variable, gender

R also usually does not allow mixing of data types for a variable, except in a:

- List – as a one dimensional vector, e.g. c("vancomycin", 1.618, "red")
- Data-frame – as a two dimensional table with rows (observations) and columns (variables)

Lists and data-frames are treated as their own 'class' in R.

Query output from MIMIC commonly will be in the form of data tables with different data types in different columns. Therefore, R usually stores these tables as ‘data-frames’ when they are read into R.

Special Values in R

- NA – ‘not available’, usually a default placeholder for missing values.
- NAN – ‘not a number’, only applying to numeric vectors.
- NULL – ‘empty’ value or set. Often returned by expressions where the value is undefined.
- Inf – value for ‘infinity’ and only applies to numeric vectors.

Setting Working Directory

This step tells R where to read in the source files.

Command: setwd(“directory_path”)

Example: (If all data files are saved in directory “MIMIC_data_files” on the Desktop)

```
setwd("~/Desktop/MIMIC_data_files")

# List files in directory:
list.files()
## [1] "c_score_sicker.csv"         "comorbidity_scores.csv"
## [3] "demographics.csv"           "mean_arterial_pressure.csv"
## [5] "population.csv"
```

Reading in .csv Files from MIMIC Query Results

The data read into R is assigned a ‘name’ for reference later on.

Command: set_var_name <- read.csv(“filename.csv”)

Example:

```
demo <- read.csv("demographics.csv")
```

Viewing the Dataset

There are several commands in R that are very useful for getting a ‘feel’ of your datasets and see what they look like before you start manipulating them.

- View the first and last 2 rows. E.g.:

```
head(demo, 2)

##   subject_id hadm_id marital_status_descr ethnicity_descr
## 1          4   17296               SINGLE           WHITE
## 2          6   23467              MARRIED           WHITE

tail(demo, 2)

##       subject_id hadm_id marital_status_descr  ethnicity_descr
## 27624      32807   32736              MARRIED UNABLE TO OBTAIN
## 27625      32805   34884             DIVORCED            WHITE
```

- View summary statistics. E.g.:

```
summary(demo)

##    subject_id       hadm_id      marital_status_descr
##  Min.  :    3   Min.  :    1     MARRIED  :13447
##  1st Qu.: 8063   1st Qu.: 9204   SINGLE   : 6412
##  Median :16060   Median :18278   WIDOWED  : 4029
##  Mean   :16112   Mean   :18035   DIVORCED : 1623
##  3rd Qu.:24119   3rd Qu.:26762            : 1552
##  Max.  :32809   Max.  :36118   SEPARATED:  320
##                                  (Other)  :  242
##                ethnicity_descr
##  WHITE                 :19360
##  UNKNOWN/NOT SPECIFIED : 3446
##  BLACK/AFRICAN AMERICAN: 2251
##  …
```

- View structure of data set (obs = number of rows). E.g.:

```
str(demo)

## 'data.frame':    27625 obs. of  4 variables:
##  $ subject_id          : int  4 6 3 9 15 14 11 18 18 19 ...
##  $ hadm_id             : int  17296 23467 2075 8253 4819 23919 28128
24759 33481 25788 ...
##  $ marital_status_descr: Factor w/ 8 levels "","DIVORCED",..: 6 4 4
1 6 4 4 4 4 1 ...
##  $ ethnicity_descr     : Factor w/ 39 levels "AMERICAN INDIAN/ALASKA
NATIVE",..: 35 35 35 34 12 35 35 35 35 35 ...
```

- Find out the ‘class’ of a variable or dataset. E.g.:

```
class(demo)

## [1] "data.frame"
```

- View number of rows and column, or alternatively, the dimension of the dataset. E.g.:

```
nrow(demo)

## [1] 27625

ncol(demo)

## [1] 4

dim(demo)

## [1] 27625     4
```

- Calculate length of a variable. E.g.:

```
x <- c(1:10); x

##  [1]  1  2  3  4  5  6  7  8  9 10

class(x)

## [1] "integer"
```

Subsetting a Dataset and Adding New Variables/Columns

Aim: Sometimes, it may be useful to look at only some columns or some rows in a dataset/data-frame—this is called subsetting.

Let's create a simple data-frame to demonstrate basic subsetting and other command functions in R. One simple way to do this is to create each column of the data-frame separately then combine them into a dataframe later. Note the different kinds of data types for the columns/variables created, and beware that R is case-sensitive.

Examples: Note that comments appearing after the hash sign (#) will not be evaluated.

```
subject_id <- c(1:6)                                          #integer
gender <- as.factor(c("F", "F", "M", "F", "M", "M"))#factor/categorical
height <- c(1.52, 1.65, 1.75, 1.72, 1.85, 1.78)      #numeric
weight <- c(56.7, 99.6, 90.4, 85.3, 71.4, 130.5)    #numeric
data <- data.frame(subject_id, gender, height, weight)

head(data, 4)                              # View only the first 4 rows

##   subject_id gender height weight
## 1          1      F   1.52   56.7
## 2          2      F   1.65   99.6
## 3          3      M   1.75   90.4
## …

str(data)                     # Note the class of each variable/column

## 'data.frame':    6 obs. of  4 variables:
##  $ subject_id: int  1 2 3 4 5 6
##  $ gender    : Factor w/ 2 levels "F","M": 1 1 2 1 2 2
##  $ height    : num  1.52 1.65 1.75 1.72 1.85 1.78
##  $ weight    : num  56.7 99.6 90.4 85.3 71.4 ...
```

To subset or extract only e.g., weight, we can use either the dollar sign ($) after the dataset, data, or use the square brackets, []. The $ selects column with the column name (without quotation mark in this case). The square brackets [] here selected the column weight by its column number:

```
w1 <- data$weight; w1

## [1]  56.7  99.6  90.4  85.3  71.4 130.5

w2 <- data[, 4]; w2

## [1]  56.7  99.6  90.4  85.3  71.4 130.5
```

Generally one can subset a dataset by specifying the rows and column desired like this: data[row number, column number]. For example:

```
dat_sub <- data[2:4, 1:3]; dat_sub

##   subject_id gender height
## 2          2      F   1.65
## 3          3      M   1.75
## 4          4      F   1.72
```

The square brackets are useful for subsetting multiple columns or rows. Note that it is important to 'concatenate', c(), if selecting multiple variables/columns and to use quotation marks when selecting with columns names

```
h_w1 <- data[, c(3, 4)]; h_w1

##   height weight
## 1   1.52   56.7
## 2   1.65   99.6
## 3   1.75   90.4
## …

h_w2 <- data[, c("height", "weight")]; h_w2

##   height weight
## 1   1.52   56.7
## 2   1.65   99.6
## 3   1.75   90.4
## …
```

To calculate the BMI (weight/height^2) in a new column—there are different ways to do this but here is a simple method:

```
data$BMI <- data$weight/data$height^2
head(data, 4)

##   subject_id gender height weight      BMI
## 1          1      F   1.52   56.7 24.54120
## 2          2      F   1.65   99.6 36.58402
## 3          3      M   1.75   90.4 29.51837
## 4          4      F   1.72   85.3 28.83315
```

Let's create a new column, obese, for BMI > 30, as TRUE or FALSE. This also demonstrates the use of 'logicals' in R.

```
data$obese <- data$BMI > 30
head(data)

##   subject_id gender height weight      BMI obese
## 1          1      F   1.52   56.7 24.54120 FALSE
## 2          2      F   1.65   99.6 36.58402  TRUE
## 3          3      M   1.75   90.4 29.51837 FALSE
## …
```

One can also use logical vectors to subset datasets in R. A logical vector, named "ob" here, is created and then we pass it through the square brackets [] to tell R to select only the rows where the condition BMI > 30 is TRUE:

```
ob <- data$BMI > 30
data_ob <- data[ob, ];data_ob

##   subject_id gender height weight      BMI obese
## 2          2      F   1.65   99.6 36.58402  TRUE
## 6          6      M   1.78  130.5 41.18798  TRUE
```

Combining Datasets (Called Data Frames in R)

Aim: Often different variables (columns) of interest in a research question may come from separate MIMIC tables and could have been exported as separate.csv files if they were not merged via SQL queries. For ease of analysis and visualization, it is often desirable to merge these separate data frames in R on their shared ID column(s).

Occasionally, one may also want to attach rows from one data frame after rows from another. In this case, the column names and the number of columns of the two different datasets must be the same.

Examples: In general, there are a couple ways of combining columns and rows from different datasets in R:

- merge()—This function merges columns on shared ID column(s) between the data frames so the associated rows match up correctly.

  `Command: merging on one ID column, e.g.:`

```
df_merged <- merge(df1, df2, by = “column_ID_name”)
```

  `Command: merging on two ID columns, e.g.:`

```
df_merged <- merge(df1, df2, by = c(“column1”, “column2”))
```

- cbind()—This function simply ‘add’ together the columns from two data frames (must have equal number of rows). It does not match up the rows by any identifier.

  `Command: joining columns. E.g.:`

```
df_total <- cbind(df1, df2)
```

- rbind()—The function ‘row binds’ the two data frames vertically (must have the same column names).

  `Command: joining rows. E.g.:`

```
df_total <- rbind(df1, df2)
```

Using Packages in R

There are many packages that make life so much easier when manipulating data in R. They need to be installed on your computer and loaded at the start of your R script before you can call the functions in them. We will introduce examples of of a couple of useful packages later in this chapter.

For now, the command for installing packages is:

```
install.packages("name_of_package_case_sensitive")
```

The command for loading the package into the R working environment:

```
library(name_of_package_case_sensitive)
```

Note—there are no quotation marks when loading packages as compared to installing; you will get an error message otherwise.

Getting Help in R

There are various online tutorials and Q&A forums for getting help in R. Stackoverflow, Cran and Quick-R are some good examples. Within the R console, a question mark, ?, followed by the name of the function of interest will bring up the help menu for the function, e.g.

```
?head
```

12.3.2 Data Integration

Aim: This involves combining the separate output datasets exported from separate MIMIC queries into a consistent larger dataset table.

To ensure that the associated observations or rows from the two different datasets match up, the right column ID must be used. In MIMIC, the ID columns could be subject_id, hadm_id, icustay_id, itemid, etc. Hence, knowing the context of what each column ID is used to identify and how they are related to each other is important. For example, subject_id is used to identify each individual patient, so includes their date of birth (DOB), date of death (DOD) and various other clinical detail and laboratory values in MIMIC. Likewise, the hospital admission ID, hadm_id, is used to specifically identify various events and outcomes from an unique hospital admission; and is also in turn associated with the subject_id of the patient who was involved in that particular hospital admission. Tables pulled from MIMIC can have one or more ID columns. The different tables exported from MIMIC may share some ID columns, which allows us to 'merge' them together, matching up the rows correctly using the unique ID values in their shared ID columns.

Examples: To demonstrate this with MIMIC data, a simple SQL query is constructed to extract some data, saved as: "population.csv" and "demographics.csv".

We will these extracted files to show how to merge datasets in R.

1. **SQL query**:

```
WITH
population AS(
SELECT subject_id, hadm_id, gender, dob, icustay_admit_age,
icustay_intime, icustay_outtime, dod, expire_flg
FROM mimic2v26.icustay_detail
  WHERE subject_icustay_seq = 1
  AND icustay_age_group = 'adult'
  AND hadm_id IS NOT NULL
)
, demo AS(
SELECT subject_id, hadm_id, marital_status_descr, ethnicity_descr
FROM mimic2v26.demographic_detail
WHERE subject_id IN (SELECT subject_id FROM population)
)

--# Extract the the datasets with each one of the following line of
codes in turn:
--SELECT * FROM population
--SELECT * FROM demo
```

Note: Remove the – in front of the SELECT command to run the query.

2. **R code: Demonstrating data integration**

Set working directory and read data files into R::

```
setwd("~/Desktop/MIMIC_data_files")
demo <- read.csv("demographics.csv", sep = ",")
pop <- read.csv("population.csv", sep = ",")
head(demo)

##   subject_id hadm_id marital_status_descr        ethnicity_descr
## 1          4   17296               SINGLE                  WHITE
## 2          6   23467              MARRIED                  WHITE
## 3          3    2075              MARRIED                  WHITE
## …
head(pop)

##   subject_id hadm_id gender                 dob icustay_admit_age
## 1          4   17296      F 3351-05-30 00:00:00          47.84414
## 2          6   23467      F 3323-07-30 00:00:00          65.94048
## 3          3    2075      M 2606-02-28 00:00:00          76.52892
## …


##        icustay_intime     icustay_outtime                 dod
expire_flg
## 1 3399-04-03 00:29:00 3399-04-04 16:46:00
N
## 2 3389-07-07 20:38:00 3389-07-11 12:47:00
N
## 3 2682-09-07 18:12:00 2682-09-13 19:45:00 2683-05-02 00:00:00
Y
## …
```

Merging pop and demo: Note to get the rows to match up correctly, we need to merge on both the subject_id and hadm_id in this case. This is because each subject/patient could have multiple hadm_id from different hospital admissions during the EHR course of MIMIC database.

```
demopop <- merge(pop, demo, by = c("subject_id", "hadm_id"))
head(demopop)

##   subject_id hadm_id gender                 dob icustay_admit_age
## 1        100     445      F 3048-09-22 00:00:00          71.94482
## 2       1000   15170      M 2442-05-11 00:00:00          69.70579
## 3      10000   10444      M 3149-12-07 00:00:00          49.67315
## …


##        icustay_intime     icustay_outtime                 dod
expire_flg
## 1 3120-09-01 11:19:00 3120-09-03 14:06:00
N
## 2 2512-01-25 13:16:00 2512-03-02 06:05:00 2512-03-02 00:00:00
Y
## 3 3199-08-09 09:53:00 3199-08-10 17:43:00
N
## …

##   marital_status_descr        ethnicity_descr
## 1              WIDOWED  UNKNOWN/NOT SPECIFIED
## 2              MARRIED  UNKNOWN/NOT SPECIFIED
## 3                          HISPANIC OR LATINO
## 4              MARRIED BLACK/AFRICAN AMERICAN
## 5              MARRIED                  WHITE
## 6            SEPARATED BLACK/AFRICAN AMERICAN
```

As you can see, there are still multiple problems with this merged database, for example, the missing values for 'marital_status_descr' column. Dealing with missing data is explored in Chap. 13.

12.3.3 Data Transformation

Aim: To transform the presentation of data values in some ways so that the new format is more suitable for the subsequent statistical analysis. The main processes involved are normalization, aggregation and generalization (See part 1 for explanation).

Examples: To demonstrate this with a MIMIC database example, let us look at a table generated from the following simple SQL query, which we exported as "comorbidity_scores.csv".

The SQL query selects all the patient comorbidity information from the mimic2v26.comorbidity_scores table on the condition of (1) being an adult, (2) in his/her first ICU admission, and (3) where the hadm_id is not missing according to the mimic2v26.icustay_detail table.

1. **SQL query:**

```
SELECT *
FROM mimic2v26.comorbidity_scores
WHERE subject_id IN (SELECT subject_id
        FROM mimic2v26.icustay_detail
        WHERE subject_icustay_seq = 1
                AND icustay_age_group = 'adult'
                AND hadm_id IS NOT null)
```

2. **R code: Demonstrating data transformation**:

```
setwd("~/Desktop/MIMIC_data_files")
c_scores <- read.csv("comorbidity_scores.csv", sep = ",")
```

Note the 'class' or data type of each column/variable and the total number of rows (obs) and columns (variables) in c_scores:

```
str(c_scores)

## 'data.frame':    27525 obs. of  33 variables:
##  $ subject_id              : int  2848 21370 2026 11890 27223 27520
17928 31252 32083 9545 ...
##  $ hadm_id                 : int  16272 17542 11351 12730 32530
32724 20353 30062 32216 10809 ...
##  $ category                : Factor w/ 1 level "ELIXHAUSER": 1 1 1 1
1 1 1 1 1 1 ...
##  $ congestive_heart_failure: int  0 0 0 0 1 0 0 0 1 1 ...
##  $ cardiac_arrhythmias     : int  0 1 1 0 1 0 0 0 0 1 ...
##  $ valvular_disease        : int  0 0 0 0 1 0 0 0 0 1 ...
##  $ …
```

Here we add a column in c_scores to save the overall ELIXHAUSER. The rep() function in this case repeats 0 for nrow(c_scores) times. Function, colnames(), rename the new or last column, [ncol(c_scores)], as “ELIXHAUSER_overall”.

```
c_scores <- cbind(c_scores, rep(0, nrow(c_scores)))
colnames(c_scores)[ncol(c_scores)] <- "ELIXHAUSER_overall"
```

Take a look at the result. Note the new “ELIXHAUSER_overall” column added at the end:

```
str(c_scores)

## 'data.frame':    27525 obs. of  34 variables:
##  $ subject_id             : int  2848 21370 2026 11890 27223 27520
17928 31252 32083 9545 ...
##  $ hadm_id                : int  16272 17542 11351 12730 32530
32724 20353 30062 32216 10809 ...
##  $ category               : Factor w/ 1 level "ELIXHAUSER": 1 1 1 1
1 1 1 1 1 1 ...
##  $ congestive_heart_failure: int  0 0 0 0 1 0 0 0 1 1 ...
##  $ cardiac_arrhythmias     : int  0 1 1 0 1 0 0 0 0 1 ...
##  $ valvular_disease        : int  0 0 0 0 1 0 0 0 0 1 ...
##  $ …
```

Aggregation Step

Aim: To sum up the values of all the ELIXHAUSER comorbidities across each row. Using a ‘for loop’, for each i-th row entry in column “ELIXHAUSER_overall”, we sum up all the comorbidity scores in that row.

```
for (i in 1:nrow(c_scores)) {
  c_scores[i, "ELIXHAUSER_overall"] <- sum(c_scores[i,4:33])
}
```

Let’s take a look at the head of the resulting first and last column:

```
head(c_scores[, c(1, 34)])

##   subject_id ELIXHAUSER_overall
## 1       2848                  1
## 2      21370                  3
## 3       2026                  3
## …
```

Normalization Step

Aim: Scale values in column ELIXHAUSER_overall to between 0 and 1, i.e. in [0, 1]. Function, max(), finds out the maximum value in column ELIXHAUSER *overall. We then re-assign each entry in column ELIXHAUSER*overall as a proportion of the max_score to normalize/scale the column.

```
max_score <- max(c_scores[,"ELIXHAUSER_overall"])
c_scores[,"ELIXHAUSER_overall"] <- c_scores[ ,
"ELIXHAUSER_overall"]/max_score
```

We subset and remove all the columns in c_score, except for “subject_id”, “hadm_id”, and “ELIXHAUSER_overall”:

```
c_scores <- c_scores[, c("subject_id", "hadm_id",
"ELIXHAUSER_overall")]
head(c_scores)

##   subject_id hadm_id ELIXHAUSER_overall
## 1       2848   16272         0.09090909
## 2      21370   17542         0.27272727
## 3       2026   11351         0.27272727
## …
```

Generalization Step

Aim: Consider only the patient sicker than the average Elixhauser score. The function, which(), return the row numbers (indices) of all the TRUE entries of the logical condition set on c_scores inside the round () brackets, where the condition being the column entry for ELIXHAUSER_overall ≥ 0.5. We store the row indices information in the vector, ‘sicker’. Then we can use ‘sicker’ to subset c_scores to select only the rows/patients who are ‘sicker’ and store this information in ‘c_score_sicker’.

```
sicker <- which(c_scores[,"ELIXHAUSER_overall"]>=0.5)
c_score_sicker <- c_scores[sicker, ]
head(c_score_sicker)

##    subject_id hadm_id ELIXHAUSER_overall
## 10       9545   10809          0.5454545
## 15      12049   27692          0.5454545
## 59      29801   33844          0.5454545
## …
```

Saving the results to file: There are several functions that will do this, e.g. write.table() and write.csv(). We will give an example here:

```
write.table(c_score_sicker, file = "c_score_sicker.csv", sep = ";",
row.names = F, col.names = F)
```

If you check in your working directory/folder, you should see the new "c_score_sicker.csv" file.

12.3.4 Data Reduction

Aim: To reduce or reshape the input data by means of a more effective representation of the dataset without compromising the integrity of the original data. One element of data reduction is eliminating redundant records while preserving needed data, which we will demonstrate in Example Part 1. The other element involves reshaping the dataset into a "tidy" format, which we will demonstrate in below sections.

Examples Part 1: Eliminating Redundant Records

To demonstrate this with a MIMIC database example, we will look at multiple records of non-invasive mean arterial pressure (MAP) for each patient. We will use the records from the following SQL query, which we exported as "mean_arterial_pressure.csv".

The SQL query selects all the patient subject_id's and noninvasive mean arterial pressure (MAP) measurements from the mimic2v26.chartevents table on the condition of (1) being an adult, (2) in his/her first ICU admission, and (3) where the hadm_id is not missing according to the mimic2v26.icustay_detail table.

1. **SQL query**:

```
SELECT subject_id, value1num
FROM mimic2v26.chartevents
WHERE subject_id IN (
SELECT subject_id
    FROM mimic2v26.icustay_detail
              WHERE subject_icustay_seq = 1
              AND icustay_age_group = 'adult'
              AND hadm_id IS NOT null)
AND itemid=456
AND value1num is not null

-- Export and save the query result as "mean_arterial_pressure.csv"
```

2. **R code**:

There are a variety of methods that can be chosen to aggregate records. In this case we will look at averaging multiple MAP records into a single average MAP for each patient. Other options which may be chosen include using the first recorded value, a minimum or maximum value, etc.

For a basic example, the following code demonstrates data reduction by averaging all of the multiple records of MAP into a single record per patient. The code uses the aggregate() function:

```
setwd("~/Desktop/MIMIC_data_files")
all_maps <- read.csv("mean_arterial_pressure.csv", sep = ",")

str(all_maps)

## 'data.frame':    790174 obs. of  2 variables:
##  $ subject_id: int  4 4 4 4 4 4 4 4 3 4 ...
##  $ value1num : num  80.7 71.7 74.3 69 75 ...
```

This step averages the MAP values for each distinct subject_id:

```
avg_maps <- aggregate(all_maps, by=list(all_maps[,1]), FUN=mean,
na.rm=TRUE)

head(avg_maps)

##   Group.1 subject_id value1num
## 1       3          3  75.10417
## 2       4          4  88.64102
## 3       6          6  91.37357
## …
```

Examples Part 2: Reshaping Dataset

Aim: Ideally, we want a “tidy” dataset reorganized in such a way so it follows these 3 rules [2, 3]:

1. Each variable forms a column
2. Each observation forms a row
3. Each value has its own cell

Datasets exported from MIMIC usually are fairly “tidy” already. Therefore, we will construct our own data frame here for ease of demonstration for rule 3. We will also demonstrate how to use some common data tidying packages.

R code: To mirror our own MIMIC dataframe, we construct a dataset with a column of subject_id and a column with a list of diagnoses for the admission.

```
diag <- data.frame(subject_id = 1:6,   diagnosis = c("PNA, CHF", "DKA",
"DKA, UTI", "AF, CHF", "AF", "CHF"))
diag
##   subject_id diagnosis
## 1          1  PNA, CHF
## 2          2       DKA
## 3          3  DKA, UTI
## …
```

Note that the dataset above is not “tidy”. There are multiple categorical variables in column “diagnosis”—breaks “tidy” data rule 1. There are multiple values in column “diagnosis”—breaks “tidy” data rule 3.

There are many ways to “tidy” and reshape this dataset. We will show one way to do this by making use of R packages “splitstackshape” [5] and “tidyr” [4] to make reshaping the dataset easier.

R package example 1—“splitstackshape”:

Installing and loading the package into R console.

```
install.packages("splitstackshape")
library(splitstackshape)
```

The function, cSplit(), can split the multiple categorical values in each cell of column “diagnosis” into different columns, “diagnosis_1” and “diagnosis_2”. If the argument, direction, for cSplit() is not specified, then the function splits the original dataset “wide”.

```
diag2 <- cSplit(diag, "diagnosis", ",")
diag2

##    subject_id diagnosis_1 diagnosis_2
## 1:          1         PNA         CHF
## 2:          2         DKA          NA
## 3:          3         DKA         UTI
## …
```

One could possibly keep it as this if one is interested in primary and secondary diagnoses (though it is not strictly “tidy” yet).

Alternatively, if the direction argument is specified as “long”, then cSplit split the function “long” like so:

```
diag3 <- cSplit(diag, "diagnosis", ",", direction = "long")
diag3
##    subject_id diagnosis
## 1:          1       PNA
## 2:          1       CHF
## 3:          2       DKA
## …
```

Note diag3 is still not “tidy” as there are still multiple categorical variables under column diagnosis—but we no longer have multiple values per cell.

R package example 2—“tidyr”:

To further “tidy” the dataset, package “tidyr” is pretty useful.

```
install.packages("tidyr")
library(tidyr)
```

The aim is to split each categorical variable under column, diagnosis, into their own columns with 1 = having the diagnosis and 0 = not having the diagnosis. To do this we first construct a third column, “yes”, that hold all the 1 values initially (because the function we are going use require a value column that correspond with the multiple categories column we want to ‘spread’ out).

```
diag3$yes <- rep(1, nrow(diag3))
diag3

##    subject_id diagnosis yes
## 1:          1       PNA   1
## 2:          1       CHF   1
## 3:          2       DKA   1
## …
```

Then we can use the spread function to split each categorical variables into their own columns. The argument, fill = 0, replaces the missing values.

```
diag4 <- spread(diag3, diagnosis, yes, fill = 0)
diag4

##    subject_id AF CHF DKA PNA UTI
## 1:          1  0   1   0   1   0
## 2:          2  0   0   1   0   0
## 3:          3  0   0   1   0   1
## …
```

One can see that this dataset is now “tidy”, as it follows all three “tidy” data rules.

12.4 Conclusion

A variety of quality control issues are common when using raw clinical data collected for non-study purposes. Data pre-processing is an important step in preparing raw data for statistical analysis. Several distinct steps are involved in pre-processing raw data as described in this chapter: cleaning, integration, transformation, and reduction. Throughout the process it is important to understand the choices made in pre-processing steps and how different methods can impact the validity and applicability of study results. In the case of EHR data, such as that in the MIMIC database, pre-processing often requires some understanding of the clinical context under which data were entered in order to guide these pre-processing choices. The objective of all the steps is to arrive at a “clean” and “tidy” dataset suitable for effective statistical analyses while avoiding inadvertent introduction of bias into the data.

Take Home Messages

- Raw data for secondary analysis is frequently "messy" meaning it is not in a form suitable for statistical analysis; data must be "cleaned" into a valid, complete, and effectively organized "tidy" database that can be analyzed.
- There are a variety of techniques that can be used to prepare data for analysis, and depending on the methods use, this pre-processing step can introduce bias into a study.
- The goal of pre-processing data is to prepare the available raw data for analysis without introducing bias by changing the information contained in the data or otherwise influencing end results.

References

1. Son NH (2006) Data mining course—data cleaning and data preprocessing. Warsaw University. Available at URL http://www.mimuw.edu.pl/~son/datamining/DM/4-preprocess.pdf
2. Grolemund G (2016) R for data science—data tidying. O'Reilly Media. Available at URL http://garrettgman.github.io/tidying/
3. Wickham H (2014) J Stat Softw 59(10). Tidy Data. Available at URL http://vita.had.co.nz/papers/tidy-data.pdf
4. Wickham H (2016) Package 'tidyr'—easily tidy data with spread() and gather() functions. CRAN. Available at URL https://cran.r-project.org/web/packages/tidyr/tidyr.pdf
5. Mahto A (2014) Package 'splitstackshape'—stack and reshape datasets after splitting concatenated values. CRAN. Available at URL https://cran.r-project.org/web/packages/splitstackshape/splitstackshape.pdf

Chapter 13
Missing Data

Cátia M. Salgado, Carlos Azevedo, Hugo Proença and Susana M. Vieira

Learning Objectives

- What are the different types of missing data, and the sources for missingness.
- What options are available for dealing with missing data.
- What techniques exist to help choose the most appropriate technique for a specific dataset.

13.1 Introduction

Missing data is a problem affecting most databases and electronic medical records (EHR) are no exception. Because most statistical models operate only on complete observations of exposure and outcome variables, it is necessary to deal with missing data, either by deleting incomplete observations or by replacing any missing values with an estimated value based on the other information available, a process called imputation. Both methods can significantly effect the conclusions that can be drawn from the data.

Identifying the source of "missingness" is important, as it influences the choice of the imputation technique. Schematically, several cases are possible: (i) the value is missing because it was forgotten or lost; (ii) the value is missing because it was not applicable to the instance; (iii) the value is missing because it is of no interest to the instance. If we were to put this in a medical context: (i) the variable is measured but for some unidentifiable reason the values are not electronically recorded, e.g. disconnection of sensors, errors in communicating with the database server, accidental human omission, electricity failures, and others; (ii) the variable is not measured during a certain period of time due to an identifiable reason, for instance the patient is disconnected from the ventilator because of a medical decision;

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-43742-2_13) contains supplementary material, which is available to authorized users.

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_13

(iii) the variable is not measured because it is unrelated with the patient condition and provides no clinical useful information to the physician [1].

An important distinction must be made between data missing for identifiable or unidentified reasons. In the first case, imputing values can be inadequate and add bias to the dataset, so the data is said to be non-recoverable. On the other hand, when data is missing for unidentifiable reasons it is assumed that values are missing because of random and unintended causes. This type of missing data is classified as recoverable.

The first section of this chapter focuses on describing the theory of some commonly used methods to handle missing data. In order to demonstrate the advantages and disadvantages of the methods, their application is demonstrated in the second part of the chapter on actual datasets that were created to study the relation between mortality and insertion of indwelling arterial catheters (IAC) in the intensive care unit (ICU).

13.2 Part 1—Theoretical Concepts

In knowledge discovery in databases, data preparation is the most crucial and time consuming task, that strongly influences the success of the research. Variable selection consists in identifying a useful subset of potential predictors from a large set of candidates (please refer to Chap. 5—Data Analysis for further information on feature selection). Rejecting variables with an excessive number of missing values (e.g. >50 %) is usually a good rule of thumb, however it is not a risk-free procedure. Rejecting a variable may lead to a loss of predictive power and ability to detect statistically significant differences and it can be a source of bias, affecting the representativeness of the results. For these reasons, variable selection needs to be tailored to the missing data mechanism. Imputation can be done before and/or after variable selection.

The general steps that should be followed for handling missing data are:

- Identify patterns and reasons for missing data;
- Analyse the proportion of missing data;
- Choose the best imputation method.

13.2.1 Types of Missingness

The mechanisms by which the data is missing will affect some assumptions supporting our data imputation methods. Three major mechanisms of missingness of the data can be described, depending on the relation between observed (available) and unobserved (missing) data.

For the sake of simplicity, lets consider missingness in the univariate case. To define missingness in mathematical terms, a dataset X can be divided in two parts:

$$X = \{X_o, X_m\} \tag{1}$$

where X_o corresponds to the observed data, and X_m to the missing data, in the dataset.

For each observation we define a binary response whether or not that observation is missing:

$$R = \begin{cases} 1 & if \;\; X\,observed \\ 0 & if \;\; X\,missing \end{cases} \tag{2}$$

The missing value mechanism can be understood in terms of the probability that an observation is missing $\Pr(R)$ given the observed and missing observations, in the form:

$$\Pr(R|x_o, x_m) \tag{3}$$

The three mechanisms are subject to whether the probability of response R depends or not on the observed and/or missing values:

- **Missing Completely at Random (MCAR)**—When the missing observations are dependent on the observed and unobserved measurements. In this case the probability of an observation being missing depends only on itself, and reduces to $\Pr(R|x_o, x_m) = \Pr(R)$. As an example, imagine that a doctor forgets to record the gender of every six patients that enter the ICU. There is no hidden mechanism related to any variable and it does not depend on any characteristic of the patients.
- **Missing at Random (MAR)**—In this case the probability of a value being missing is related only to the observable data, i.e., the observed data is statistically related with the missing variables and it is possible to estimate the missing values from the observed data. This case is not completely 'random', but it is the most general case where we can ignore the missing mechanism, as we control the information upon which the missingness depends, the observed data. Said otherwise, the probability that some data is missing for a particular variable does not depend on the values of that variable, after adjusting for observed values. Mathematically the probability of missing reduces to $\Pr(R|x_o, x_m) = \Pr(R|x_o)$. Imagine that if elderly people are less likely to inform the doctor that they had had a pneumonia before, the response rate of the variable pneumonia will depend on the variable age.
- **Missing Not at Random (MNAR)**—This refers to the case when neither MCAR nor MAR hold. The missing data depends on both missing and observed values. Determining the missing mechanism is usually impossible, as it depends on unseen data. From that derives the importance of performing sensitivity analyses and test how the inferences hold under different assumptions. For example, we can imagine that patients with low blood pressure are more likely to have their blood pressure measured less frequently (the missing data for the variable "blood pressure" partially depends on the values of the blood pressure).

13.2.2 Proportion of Missing Data

The percentage of missing data for each variable (between patients) and each patient (between variables) must be computed, to help decide which variables and/or patients should be considered candidates for removal or data imputation. A crude example is shown in Table 13.1, where we might want to consider removing patient 1 and the variable "AST" from the analysis, considering that most of their values are missing.

13.2.3 Dealing with Missing Data

Overview of Methods for Handling Missing Data

The methods should be tailored to the dataset of interest, the reasons for missingness and the proportion of missing data. In general, a method is chosen for its simplicity and its ability to introduces as little bias as possible in the dataset.

When data are MCAR or MAR a researcher can ignore the reasons for missing data, which simplifies the choice of the methods to apply. In this case, any method can be applied. Nevertheless it is difficult to obtain empirical evidence about whether or not the data are MCAR or MAR. A valid strategy is to examine the sensitivity of results to the MCAR and MAR assumptions by comparing several analyses, where the differences in results across several analyses may provide some information about what assumptions may be the most relevant.

A significant body of evidence has focused on comparing the performance of missing data handling methods, both in general [2–4] and in context of specific factors such as proportion of missing data and sample size [5–7]. More detailed technical aspects, and application of these methods in various fields can also be found in the works of Jones and Little [8, 9].

In summary, the most widely used methods fall into three main categories, which are described in more detail below.

1. Deletion methods (listwise deletion, i.e. complete-case analysis, pairwise deletion, i.e. available-case analysis)
2. Single Imputation Methods (mean/mode substitution, linear interpolation, Hot deck and cold deck)
3. Model-Based Methods (regression, multiple imputation, k-nearest neighbors)

Table 13.1 Examples of missing data in EHR

	Gender	Glucose	AST	Age
Patient 1	?	120	?	?
Patient 2	M	105	?	68
Patient 3	F	203	45	63
Patient 4	M	145	?	42
Patient 5	M	89	?	80

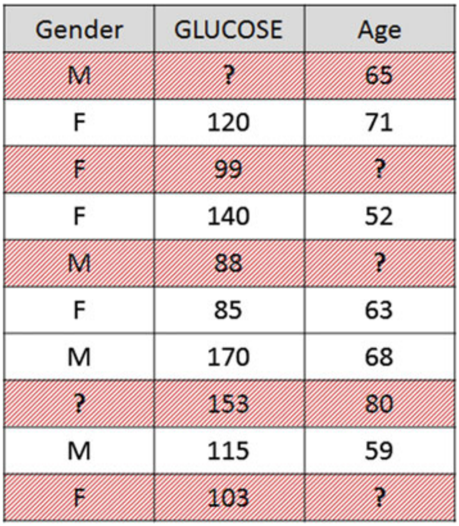
Deletion Methods

The simplest way to deal with missing data is to discard the cases or observations that have missing values. In general, case deletion methods lead to valid inferences only for MCAR [10]. There are three ways of doing this: complete-case analysis; available-case analysis; and weighting methods.

Complete-Case Analysis (Listwise Deletion)

In complete case analysis, all the observations with at least one missing variable are discarded (Fig. 13.1).

The principal assumption is that the remaining subsample is representative of the population, and will thus not bias the analysis towards a subgroup. This assumption is rather restrictive and assumes a MCAR mechanism. Listwise deletion often produces unbiased regression slope estimates, as long as missingness is not a function of the outcome variable. The biggest advantage of this method is its simplicity, it is always reasonable to use it when the number of discarded observations is relatively small when compared to the total. Its main drawbacks are the reduced statistical power (because it reduces the number of samples *n,* the estimates will have larger standard errors), waste of information, and possible bias of the analysis specially if data is not MCAR.



Gender	GLUCOSE	Age
M	?	65
F	120	71
F	99	?
F	140	52
M	88	?
F	85	63
M	170	68
?	153	80
M	115	59
F	103	?

Fig. 13.1 Example of complete-case deletion. Cases highlighted in *red* are discarded

Available-Case Analysis

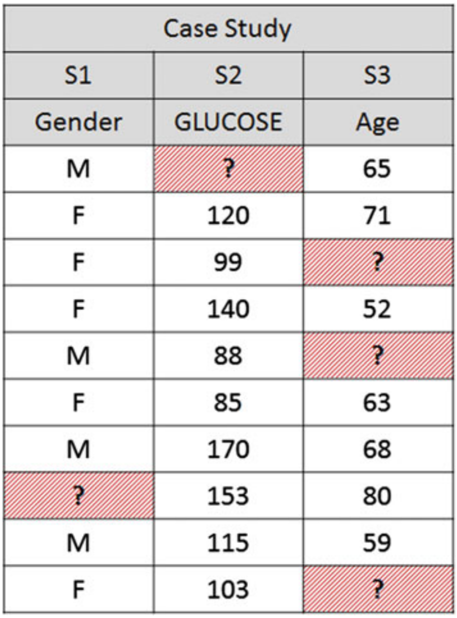
The available-case method discards data only in the variables that are needed for a specific analysis. For example, if only 4 out of 20 variables are needed for a study, this method would only discard the missing observations of the 4 variables of interest. In Fig. 13.2, imagine that each one of the three represented variables would be used for a different analysis. The analysis is performed using all cases in which the variables of interest are present. Even though this method has the ability to preserve more information, the populations of each analysis would be different and possibly non-comparable.

Weighting-Case Analysis

Weighting is a way of weighting the complete-cases by modelling the missingness in order to reduce the bias introduced in the available-case.

Single-Value Imputation

In single imputation, missing values are filled by some type of "predicted" values [9, 11]. Single imputation ignores uncertainty and almost always underestimates the variance. Multiple imputation overcomes this problem, by taking into account both within—and between—imputation uncertainty.



Case Study		
S1	S2	S3
Gender	GLUCOSE	Age
M	?	65
F	120	71
F	99	?
F	140	52
M	88	?
F	85	63
M	170	68
?	153	80
M	115	59
F	103	?

Fig. 13.2 Example of available-case deletion. If each variable is used for separate analyses, only the cases in which the variable of interest is missing are discarded

Mean and Median

The simplest imputation method is to substitute missing values by the mean or the median of that variable. Using the median is more robust in the presence of outliers in the observed data. The main disadvantages are that (1) it reduces variability, thereby lowering the estimate errors compared to deletion approaches, and (2) it disregards the relationship between variables, decreasing therefore their correlation. While this method diminishes the bias of using a non-representative sample, it introduces other bias.

Linear Interpolation

This method is particularly suitable for time-series. In linear interpolation, a missing value is computed by interpolating the values of the previous and next available measurements for the patient. For example, if the natremia changes from 132 to 136 mEq/L in 8 h, one can reasonably assume that its value was close to 134 mEq/L at midpoint.

Hot Deck and Cold Deck

In the hot deck method, a missing attribute value is replaced with a value from an estimated distribution of the current data. It is especially used in survey research [9]. Hot deck is typically implemented in two stages. First, the data is partitioned into clusters, and then each instance with missing data is associated with one cluster. The complete cases in a cluster are used to fill in the missing values. This can be done by calculating the mean or mode of the attribute within a cluster. Cold deck imputation is similar to hot deck, except that the data source is different from the current dataset. Hot-deck imputation replaces the missing data by realistic values that preserve the variable distribution. However it underestimates the standard errors and the variability [12].

Last Observation Carried Forward

Sometimes called “sample-and-hold” method [13]. The last value carried forward method is specific to longitudinal designs. This technique imputes the missing value with the last available observation of the individual. This method makes the assumption that the observation of the individual has not changed at all since the last measured observation, which is often unrealistic [14].

Model-Based Imputation

In model-based imputation, a predictive model is created to estimate values that will substitute the missing data. In this case, the dataset is divided into two subsets: one with no missing values for the variable under evaluation (used for training the model) and one containing missing values, that we want to estimate. Several modeling methods can be used such as: regression, logistic regression, neural networks and other parametric and non-parametric modeling techniques. There are two main drawbacks in this approach: the model estimates values are usually more well-behaved than the true values, and the models perform poorly if the observed and missing variables are independent.

Linear Regression

In this model, all the available variables are used to create a linear regression model using the available observations of the variable of interest as output. The advantages of this method is that it takes into account the relationship between variables, unlike the mean/median imputation. The disadvantages are that it overestimates the model fit and the correlation between the variables, as it does not take into account the uncertainty in the missing data and underestimates variances and covariances. A method that was created to introduce uncertainty is the stochastic linear regression (see below).

The case of multivariate imputation is more complex as missing values exist for several variables, which do not follow the same pattern of missingness through the observations. The method used is a multivariate extension of the linear model and relies on an iterative process carried until convergence.

Stochastic Regression

Stochastic regression imputation aims to reduce the bias by an extra step of augmenting each predicted score with a residual term. This residual term is normally distributed with a mean of zero and a variance equal to the residual variance from the regression of the predictor on the target. This method allows to preserve the variability in the data and unbiased parameter estimates with MAR data. However, the standard error tends to be underestimated, because the uncertainty about the imputed values is not included, which increases the risk of type I error [15].

Multiple-Value Imputation

Multiple Imputation (MI) is a powerful statistical technique developed by Rubin in the 1970s for analysing datasets containing missing values [7, 16]. It is a Monte Carlo technique that requires 3 steps (Fig. 13.3).

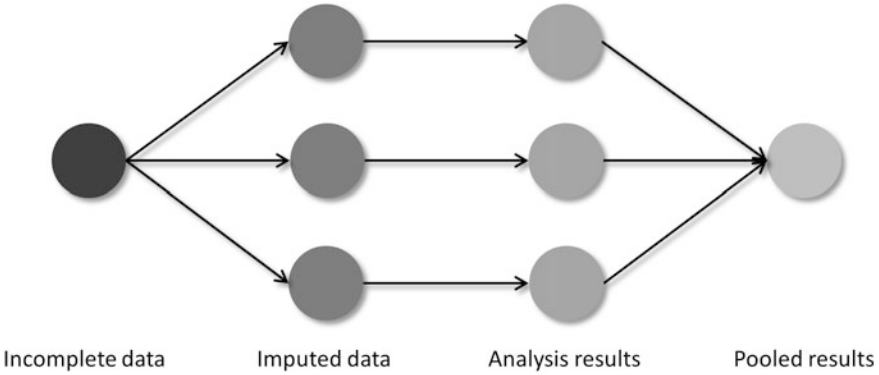


Fig. 13.3 The concept of multiple imputation, with $M = 3$

- Imputation, where the missing values are filled in using any method of choice, leading to $M \geq 2$ completed datasets (5–10 is generally sufficient) [10]. In these M multiply-imputed datasets, all the observed values are the same, but the imputed values are different, reflecting the uncertainty about imputation [10].
- Analysis: each of the M completed datasets is analysed (e.g. a logistic regression classifier for mortality prediction is built), which gives M analyses.
- Pooling: the M analyses are integrated into a final result, for example by computing the mean (and 95 % CI) of the M analyses.

K-Nearest Neighbors

K-nearest neighbors (kNN) can be used for handling missing values. Here, they will be filled with the mean of the k values coming from the k most similar complete observations. The similarity of two observations is determined, after normalization of the dataset, using a distance function which can be Euclidean, Manhattan, Mahalanobis, Pearson, etc. The main advantage of the kNN algorithm is that given enough data it can predict with a reasonable accuracy the conditional probability distribution around a point and thus make well informed estimations. It can predict qualitative and quantitative (discrete and continuous) attributes. Another advantage of this method is that the correlation structure of the data is taken into consideration. The choice of the k-value is very critical. A higher value of k would include attributes which are significantly different from our target observation, while lower value of k implies missing out of significant attributes.

13.2.4 *Choice of the Best Imputation Method*

Different imputation methods are expected to perform differently on various datasets. We describe here a generic and simple method that can be used to evaluate the performance of various imputation methods on your own dataset, in order to help selecting the most appropriate method. Of note, this simple approach does not test the effect of deletion methods. A more complex approach is described in the case study below, in which the performance of a predictive model is tested on the dataset completed by various imputation methods.

Here is how to proceed:

1. Use a sample of your own dataset that does not contain any missing data (will serve as ground truth).
2. Introduce increasing proportions of missing data at random (e.g. 5–50 % in 5 % increments).
3. Reconstruct the missing data using the various methods.
4. Compute the sum of squared errors between the reconstructed and the original data, for each method and each proportion of missing data.
5. Repeat steps 1–4 a number of times (10 times for example) and compute the average performance of each method (average SSE).
6. Plot the average SSE versus proportion of missing data (1 plot per imputation method), similarly to the example shown in Fig. 13.4.

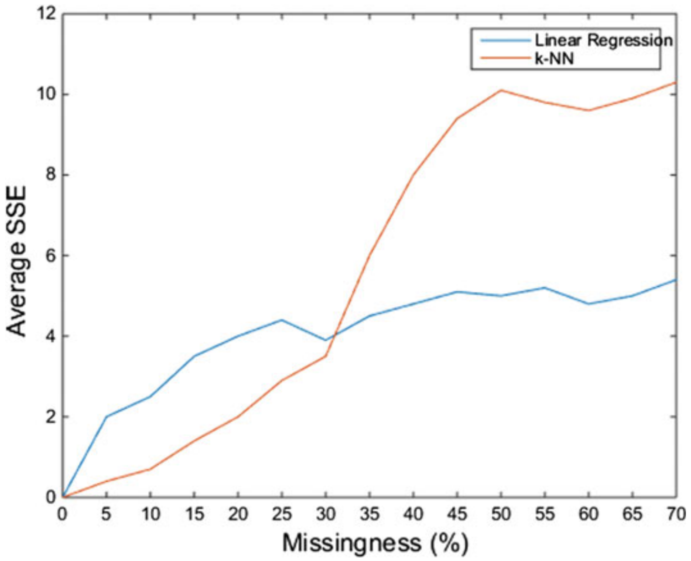


Fig. 13.4 Average SSE between original and reconstructed data, for various levels of missingness and 2 imputation methods (data only for illustrative purposes)

7. Choose the method that performs best at the level of missing data in your dataset. E.g. if your data had 10 % of missing data, you would want to pick k-NN; at 40 % linear regression performs better (made-up data, for illustrative purpose only).

13.3 Part 2—Case Study

In this section, various imputation methods will be applied to two “real world” clinical datasets used in a study that investigated the effect of inserting an indwelling arterial catheter (IAC) in patients with respiratory failure. Two datasets are used, and include patients that received an IAC (IAC group) and patients that did not (non-IAC). Each dataset is subdivided into 2 classes, with class 1 corresponding to patients that died within 28 days and class 0 to survivors. The proportion of missing data and potential reasons for missingness are discussed first. The following analyses were then carried out:

1. Various proportions of missing data at random were inserted into the variable “age”, then imputed using the various methods described above. The distribution of the imputed observations was compared to the original distribution for all the methods.
2. The performance of imputed datasets with different degrees of missingness was tested on a predictive model (logistic regression to predict mortality), first for univariate missing data (the variable age), then for all the variables (multivariate).

The code used to generate the analyses and the figures is provided in the in the accompanying R functions document.

13.3.1 Proportion of Missing Data and Possible Reasons for Missingness

Table 13.2 shows the proportion of missing data in some of the variables of the datasets. 26 variables represent the subset that was considered for testing the different imputation methods, and were selected based on the assumption that missing data occurring in these variables is recoverable.

Since IAC are mainly used for continuous hemodynamic monitoring and for arterial blood sampling for blood gas analysis, we can expect a higher percentage of missing data in blood gas-related variables in the non-IAC group. We can also expect that patient diagnoses are often able to provide an explanation for the lack of specific laboratory results: if a certain test is not ordered because it will most likely provide no clinical insight, a missing value will occur; it is fair to estimate that such

Table 13.2 Missing data in some of the variables of the IAC and non-IAC datasets

	IAC		Non-IAC	
	# points	%	# points	%
Arterial line time day	0	0	792	100
Hospital length of stay	0	0	0	0
Age	0	0	0	0
Gender	0	0	0	0
Weight first	39	3.96	71	8.96
SOFA first	2	0.20	4	0.51
Hemoglobin first	2	0.20	5	0.63
Bilirubin first	418	42.48	365	46.09
…				

value lies within a normal range. In both cases, the fact that data is missing contains information about the response, thus it is MNAR. Body mass index (BMI) has a relatively high percentage of missing data. Assuming that this variable is calculated automatically from the weight and height of patients, we can conclude that this data is MAR: because the height and/or weight are missing, BMI cannot be calculated. If the weight is missing because someone forgot to introduce it into the system then it is MCAR. Besides the missing data mechanism, it is also important to consider the sample distribution in each variable, as some imputation methods assume specific data distributions, usually the normal distribution.

13.3.2 Univariate Missingness Analysis

In this section, the specific influence of each imputation method will be explored for the variable age, using all the other variables. Two different levels of missingness (20 and 40 %) were artifically introduced in the datasets. The original dataset represents the ground truth, to which the imputed datasets were compared using frequency histograms.

Complete-Case Analysis

The complete-case analysis method discards all the incomplete observations with at least one missing value. The distribution of the "imputed" dataset is going to be equal to the original dataset minus the observations that have a missing value in variable age. Figure 13.5 shows an example of the distribution of the variable age in the IAC group.

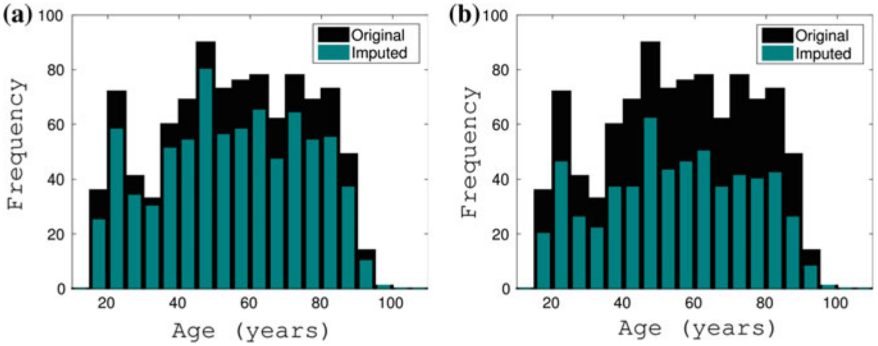


Fig. 13.5 Histogram of variable age in the IAC group before and after univariate complete case method

This method is only exploitable when there is a small percentage of missing data. This method does not require any assumption in the distribution of the missing data, besides that the complete cases should be representative of the original population, which is difficult to prove.

Single Value Imputation

Mean and Median Imputation

Mean and median methods are very crude imputation techniques, which ignore the relationship between age and the other variables and introduce a heavy bias towards the mean/median values. These simple methods allow us to better understand the biasing effect, something that is obvious in the examples Fig. 13.6.

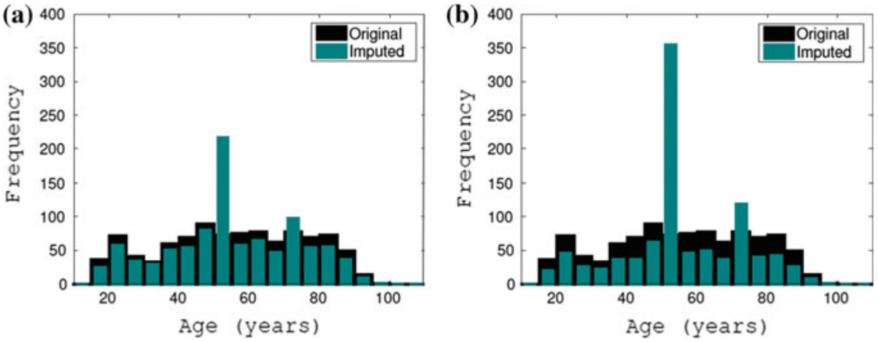


Fig. 13.6 Histogram of variable age in the IAC group before (original) and after (imputed) mean for univariate imputation

Linear Regression Imputation

The linear regression method imputes most of the data at the center of the distribution (example in Fig. 13.7). The extremities of the distribution are not well modeled and are easily ignored. This is due to two features of this technique: first, the assumption that the linear regression is a good fit to the data, and second, the assumption that the missing data lays over the regression line, bending the reality to fit the deterministic nature of the model. Compared to the mean/median imputation, the linear regression assumes a relation between the variables, however it overestimates this relation by assuming that the missing points are over the regression line. The model assumes that the percentage of variance explained is 100 %, thus it underestimates variability.

Stochastic Linear Regression Imputation

The stochastic linear regression is an attempt to loosen the deterministic assumption of the linear regression. In this case, the distribution of the imputed data fits better the original data than previous methods (Fig. 13.8). This method can introduce impossible values, such as negative age. It is a first step to model the uncertainty present in the dataset that represents a trade-off between the precision of the values and the uncertainty introduced by the missing data.

K-Nearest Neighbors

We limit the demonstration to the case where $k = 1$. In the extreme case where all neighbors are used without weights, this method converges to the mean imputation.

Figure 13.9 demonstrates that this method introduces in our particular dataset a huge bias towards the central value. The reason for this arises from the fact that

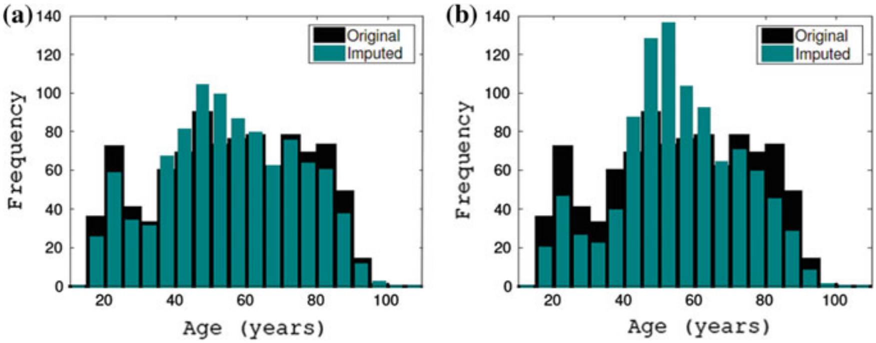


Fig. 13.7 Histogram of the variable age in the IAC group before (original) and after (imputed) linear for univariate imputation

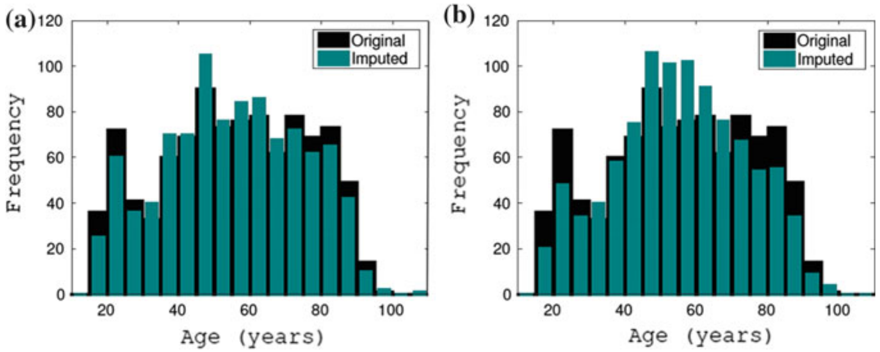


Fig. 13.8 Histogram of variable age in the IAC group before (original) and after (imputed) stochastic linear for univariate imputation

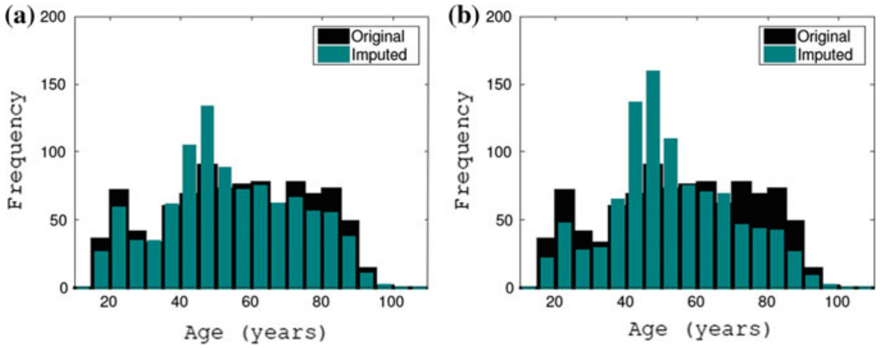


Fig. 13.9 Histogram of variable age in the IAC group before (original) and after (imputed) KNN for univariate imputation

almost half of the variables are binary, which end up having a much higher weight on the distances than continuous variables (which are always less than 1, due to the unitary normalization performed in data pre-processing). Computations with kNN increase in quality with the number of observations in the dataset, and indeed this method is very powerful given the right conditions.

Multiple Imputation

Multiple imputation with linear regression and multivariate normal regression are extensions of the single imputation methods of the same name and use sampling to create multiple different datasets, that represent different possibilities of what might be the original dataset. These methods allow a better modeling of the uncertainty present in the missing values and are, usually, more solid in terms of statistical

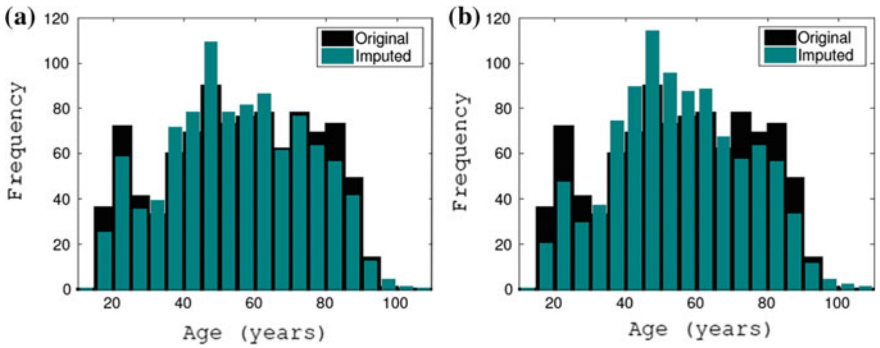


Fig. 13.10 Histogram of variable age in the IAC group before (original) and after (imputed) multiple imputation multivariate normal regression for univariate imputation

properties and results. We chose to work with 10 datasets, which were averaged so that the graphical representation would look similar to the previous methods.

Multivariate normal regression

Multiple imputation multivariate normal distribution gave more importance to the values of the center of the distribution (Fig. 13.10). The main assumption of this method is that the data follows a multivariate normal distribution, something that is not completely true for this dataset, which contains numerous binary variables. Nonetheless, even in the presence of categorical variables and distributions that are not strictly normal, it should perform reasonably well [10, 19]. The multiple imputation method enhances the modeling of uncertainty by adding a bootstrap sampling to the expectation maximization algorithm, giving raise to better predictions of the possible missing data by considering multiple possibilities of the original data. Obviously, when averaging the data for histogram representation, some of that richness is lost. Nonetheless, the quality of the regression is obvious when compared to the previous methods.

Linear regression

The multiple imputation linear regression method uses all the variables except the target variable (age) to estimate the missing data of this last variable. The data is modelled using linear regression and Gibbs sampling. Figure 13.11 demonstrates that this represents by far the most accurate imputation method in this particular dataset.

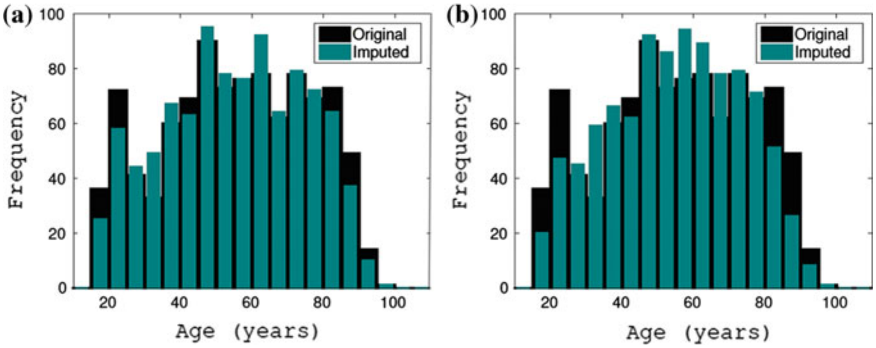


Fig. 13.11 Histogram of variable age in the IAC group before (original) and after (imputed) multiple imputation generalized regression for univariate imputation

13.3.3 Evaluating the Performance of Imputation Methods on Mortality Prediction

This test aims to assess the generalization capabilities of the models constructed using imputed data, and check their performance by comparing them to the original data. All the methods described previously were used to reconstruct a sample of both IAC and non-IAC datasets, with increasing proportions of missing data at random, first only on the variable age (univariate), then on all the variables in the dataset (multivariate). A logistic regression model was built on the reconstructed data and tested on a sample of the original data (that does not contain imputations or missing data).

The performance of the models is evaluated in terms of area under the receiver operating characteristic curve (AUC), accuracy (correct classification rate), sensitivity (true positive classification rate—TPR, also known as recall), specificity (true negative classification rate—TNR) and Cohen's kappa. All the methods were compared against a reference logistic regression that was fitted with the original data without missingness. The results were averaged over a 10-fold cross validation and the AUC results are presented graphically.

The influence of one variable has a limited effect, even if age is the variable most correlated with mortality (Fig. 13.12). At most, the AUC decreased from 0.84 to 0.81 for IAC and from 0.90 to 0.87 for the non-IAC case, if we exclude the complete-case analysis method that performs poorly from the beginning. For lower values of missingness (less than 50 %), all the other models perform similarly. Among univariate techniques, the methods that performed the best on both datasets are the two multiple imputation methods, namely the linear regression and the multivariate normal distribution, and the one-nearest neighbors algorithm. In the case of univariate missingness, the nearest neighbors reveals to be a good estimator if several complete observations exist, as it is the case. With increasing of the

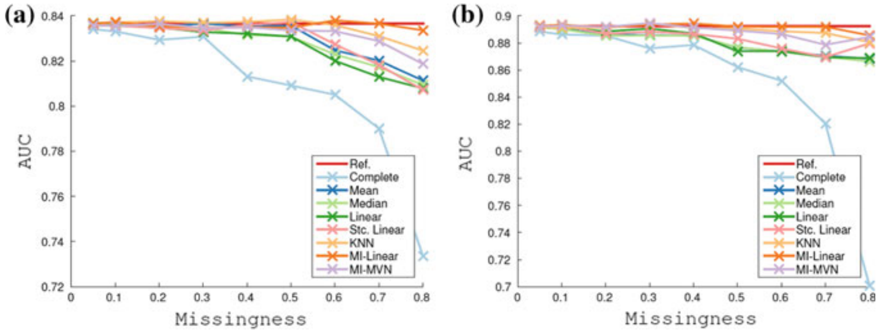


Fig. 13.12 Mean AUC performance of the logistic regression models modelled with different imputation methods for different degrees of univariate missingness of the Age variable

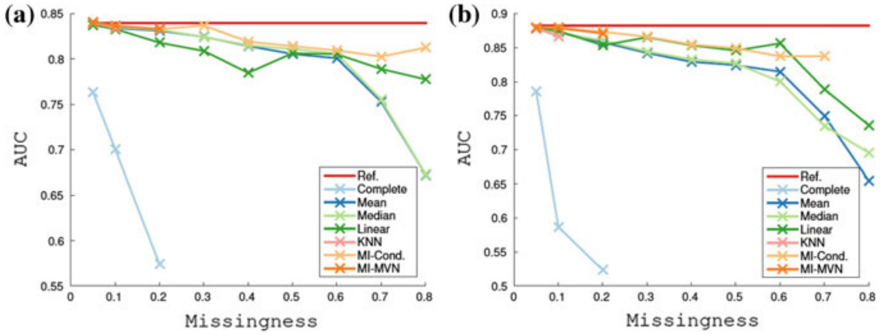


Fig. 13.13 Mean AUC of the logistic regression models for different degrees of multivariate missingness

missingness, the simpler methods introduced more bias in the modeling of the datasets.

The quality of the imputation methods was also evaluated in the presence of multivariate missingness with an uniform probability in all variables (Fig. 13.13). It has to be noted that obtaining results for more than 40 % of missingness in all the variables is quite infeasible in most cases, and there are no assurances of good performances with any of the methods. Some methods were not able to perform complete imputations over a certain degree of missingness (e.g. the complete-case analysis stopped having enough observations after 20 % of missingness).

Overall, and quite surprisingly, the methods had a reasonable performance even for 80 % of missingness in every variable. The reason behind this is that almost half of the variables are binary, and because of their relation with the output, reconstructing them from frequent values in each class is usually the best guess. The decrease in AUC was due to a decrease in the sensitivity, as the specificity values remained more or less unchanged with the increase in missingness. The method that performed the best overall in terms of AUC was the multiple imputation linear regression. In IAC it achieved a minimum value of AUC of 0.81 at 70 % of missingness, corresponding to a reference AUC of 0.84 and in non-IAC it achieved an AUC of 0.85 at 70 % of missingness, close to the reference AUC of 0.89.

13.4 Conclusion

Missing data is a widespread problem in EHR due to the nature of medical information itself, the massive amounts of data collected, the heterogeneity of data standards and recording devices, data transfers and conversions, and finally Human errors and omissions. When dealing with the problem of missing data, just like in many other domains of data mining, there is no one-size-fits-all approach, and the data scientist should ultimately rely on robust evaluation tools when choosing an imputation method to handle missing values in a particular dataset.

Take-Home Messages

- Always evaluate the reasons for missingness: is it MCAR/MAR/MNAR?
- What is the proportion of missing data per variable and per record?
- Multiple imputation approaches generally perform better than other methods.
- Evaluation tools must be used to tailor the imputation methods to a particular dataset.

References

1. Cismondi F, Fialho AS, Vieira SM, Reti SR, Sousa JMC, Finkelstein SN (2013) Missing data in medical databases: impute, delete or classify? Artif Intell Med 58(1):63–72
2. Peng CY, Harwell MR, Liou SM, Ehman LH (2006) Advances in missing data methods and implications for educational research
3. Peugh JL, Enders CK (2004) Missing data in educational research: a review of reporting practices and suggestions for improvement. Rev Educ Res 74(4):525–556
4. Young W, Weckman G, Holland W (2011) A survey of methodologies for the treatment of missing values within datasets: limitations and benefits. Theor Issues Ergon Sci 12(1):15–43

5. Alosh M (2009) The impact of missing data in a generalized integer-valued autoregression model for count data. J Biopharm Stat 19(6):1039–1054
6. Knol MJ, Janssen KJM, Donders ART, Egberts ACG, Heerdink ER, Grobbee DE, Moons KGM, Geerlings MI (2010) Unpredictable bias when using the missing indicator method or complete case analysis for missing confounder values: an empirical example. J Clin Epidemiol 63(7):728–736
7. Little RJA, Rubin DB (2002) Missing data in experiments. In: Statistical analysis with missing data. Wiley, pp 24–40
8. Jones MP (1996) Indicator and stratification methods for missing explanatory variables in multiple linear regression. J Am Stat Assoc 91(433):222–230
9. Little RJA (2016) Statistical analysis with missing data. Wiley, New York
10. Schafer JL (1999) Multiple imputation: a primer. Stat Methods Med Res 8(1):3–15
11. de Waal T, Pannekoek J, Scholtus S (2011) Handbook of statistical data editing and imputation. Wiley, New York
12. Roth PL (1994) Missing data: a conceptual review for applied psychologists. Pers Psychol 47 (3):537–560
13. Hug CW (2009) Detecting hazardous intensive care patient episodes using real-time mortality models. Thesis, Massachusetts Institute of Technology
14. Wood AM, White IR, Thompson SG (2004) Are missing outcome data adequately handled? A review of published randomized controlled trials in major medical journals. Clin Trials 1 (4):368–376
15. Enders CK (2010) Applied missing data analysis, 1st edn. The Guilford Press, New York
16. Rubin DB (1988) An overview of multiple imputation. In: Proceedings of the survey research section, American Statistical Association, pp 79–84
17. Saeed M, Villarroel M, Reisner AT, Clifford G, Lehman L-W, Moody G, Heldt T, Kyaw TH, Moody B, Mark RG (2011) Multiparameter intelligent monitoring in intensive care II (MIMIC-II): a public-access intensive care unit database. Crit Care Med 39(5):952–960
18. Scott DJ, Lee J, Silva I, Park S, Moody GB, Celi LA, Mark RG (2013) Accessing the public MIMIC-II intensive care relational database for clinical research. BMC Med Inform Decis Mak 13(1):9
19. Schafer JL, Olsen MK (1998) Multiple imputation for multivariate missing-data problems: a data analyst's perspective. Multivar Behav Res 33(4):545–571

Chapter 14
Noise Versus Outliers

Cátia M. Salgado, Carlos Azevedo, Hugo Proença
and Susana M. Vieira

Learning Objectives

- What common methods for outlier detection are available.
- How to choose the most appropriate methods.
- How to assess the performance of an outlier detection method and how to compare different methods.

14.1 Introduction

An outlier is a data point which is different from the remaining data [1]. Outliers are also referred to as *abnormalities*, *discordants*, *deviants* and *anomalies* [2]. Whereas noise can be defined as mislabeled examples (class noise) or errors in the values of attributes (attribute noise), outlier is a broader concept that includes not only errors but also discordant data that may arise from the natural variation within the population or process. As such, outliers often contain interesting and useful information about the underlying system. These particularities have been exploited in fraud control, intrusion detection systems, web robot detection, weather forecasting, law enforcement and medical diagnosis [1], using in general methods of supervised outlier detection (see below).

Within the medical domain in general, the main sources of outliers are equipment malfunctions, human errors, anomalies arising from patient specific behaviors and natural variation within patients. Consider for instance an anomalous blood test result. Several reasons can explain the presence of outliers: severe pathological states, intake of drugs, food or alcohol, recent physical activity, stress, menstrual cycle, poor blood sample collection and/or handling. While some reasons may point to the existence of patient-specific characteristics discordant with the "average"

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-43742-2_14) contains supplementary material, which is available to authorized users.

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_14

patient, in which case the observation being an outlier provides useful information, other reasons may point to human errors, and hence the observation should be considered for removal or correction. Therefore, it is crucial to consider the causes that may be responsible for outliers in a given dataset before proceeding to any type of action.

The consequences of not screening the data for outliers can be catastrophic. The negative effects of outliers can be summarized in: (1) increase in error variance and reduction in statistical power; (2) decrease in normality for the cases where outliers are non-randomly distributed; (3) model bias by corrupting the true relationship between exposure and outcome [3].

A good understanding of the data itself is required before choosing a model to detect outliers, and several factors influence the choice of an outlier identification method, including the type of data, its size and distribution, the availability of ground truth about the data, and the need for interpretability in a model [2]. For example, regression-based models are better suited for finding outliers in linearly correlated data, while clustering methods are advisable when the data is not linearly distributed along correlation planes. While this chapter provides a description of some of the most common methods for outlier detection, many others exist.

Evaluating the effectiveness of an outlier detection algorithm and comparing the different approaches is complex. Moreover, the ground-truth about outliers is often unavailable, as in the case of unsupervised scenarios, hampering the use of quantitative methods to assess the effectiveness of the algorithms in a rigorous way. The analyst is left with the alternative of qualitative and intuitive evaluation of results [2]. To overcome this difficulty, we will use in this chapter logistic regression models to investigate the performance of different outlier identification techniques in the medically relevant case study.

14.2 Part 1—Theoretical Concepts

Outlier identification methods can be classified into supervised and unsupervised methods, depending on whether prior information about the abnormalities in the data is available or not. The techniques can be further divided into univariable and multivariable methods, conditional on the number of variables considered in the dataset of interest.

The simplest form of outlier detection is extreme value analysis of unidimensional data. In this case, the core principle of discovering outliers is to determine the statistical tails of the underlying distribution and assume that either too large or too small values are outliers. In order to apply this type of technique to a multidimensional dataset, the analysis is performed one dimension at a time. In such a multivariable analysis, outliers are samples which have unusual combinations with other samples in the multidimensional space. It is possible to have outliers with reasonable marginal values (i.e. the value appears normal when confining oneself to one dimension), but due to linear or non-linear combinations of multiple attributes

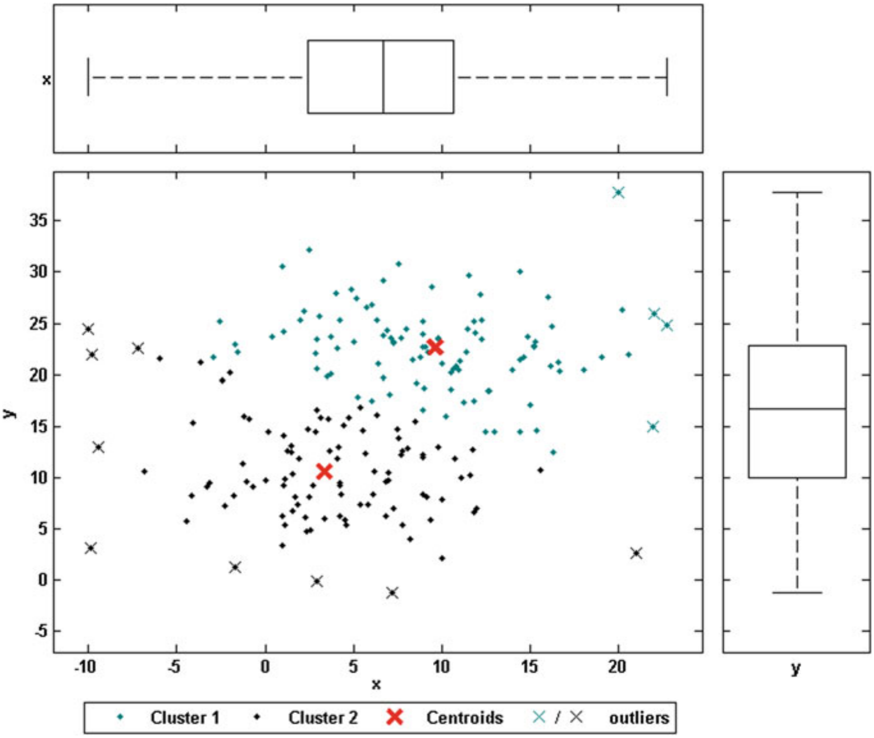


Fig. 14.1 Univariable (*boxplots*) versus multivariable (*scatter plot*) outlier investigation

these observations unveil unusual patterns in regards to the rest of the population under study.

To better understand this, the Fig. 14.1 provides a graphical example of a scenario where outliers are only visible in a 2-dimensional space. An inspection of the boxplots will reveal no outliers (no data point above and below 1.5 IQR (the interquartile range, refer to Chap. 15—Exploratory Data Analysis), a widely utilized outlier identification method), whereas a close observation of the natural clusters present in data will uncover irregular patterns. Outliers can be identified by visual inspection, highlighting data points that seem to be relatively out of the inherent 2-D data groups.

14.3 Statistical Methods

In the field of statistics, the data is assumed to follow a distribution model (e.g., normal distribution) and an instance is considered an outlier if it deviates significantly from the model [2, 4]. The use of normal distributions simplifies the analysis, as most of the existing statistical tests, such as the Z-score, can be directly interpreted in terms of probabilities of significance. However, in many real world datasets the underlying distribution of the data is unknown or complex. Statistical tests still provide a good approximation of outlier scores, but results of the tests need to be interpreted carefully and cannot be expressed statistically [2]. The next sections describe some of the most widely used statistical tests for outliers identification.

14.3.1 Tukey's Method

Quartiles are the values that divide an array of numbers into quarters. The (IQR) is the distance between the lower (Q1) and upper (Q3) quartiles in the boxplot, that is $IQR = Q3 - Q1$. It can be used as a measure of how spread out the values are. Inner "fences" are located at a distance of 1.5 IQR below Q1 and above Q3, and outer fences at a distance of 3 IQR below Q1 and above Q3 [5]. A value between the inner and outer fences is a possible outlier, whereas a value falling outside the outer fences is a probable outlier. The removal of all possible and probable outliers is referred to as the Interquartile (IQ) method, while in Tukey's method only the probable outliers are discarded.

14.3.2 Z-Score

The Z-value test computes the number of standard deviations by which the data varies from the mean. It presents a reasonable criterion for the identification of outliers when the data is normally distributed. It is defined as:

$$z_i = \frac{x_i - \overline{x}}{s} \tag{14.1}$$

where $\overline{x}$ and s denote the sample mean and standard deviation, respectively. In cases where mean and standard deviation of the distribution can be accurately estimated (or are available from domain knowledge), a good "rule of thumb" is to consider values with $|z_i| \geq 3$ as outliers. Of note, this method is of limited value for small datasets, since the maximum z-score is at most $n - 1/\sqrt{n}$ [6].

14.3.3 Modified Z-Score

The estimators used in the z-Score, the sample mean and sample standard deviation, can be affected by the extreme values present in the data. To avoid this problem, the modified z-score uses the median $\widetilde{x}$ and the median absolute deviation (MAD) instead of the mean and standard deviation of the sample [7]:

$$M_i = \frac{0.6745(x_i - \widetilde{x})}{MAD} \tag{14.2}$$

where

$$MAD = median\{|x_i - \widetilde{x}|\} \tag{14.3}$$

The authors recommend using modified z-scores with $|M_i| \geq 3.5$ as potential outliers. The assumption of normality of the data still holds.

14.3.4 Interquartile Range with Log-Normal Distribution

The statistical tests discussed previously are specifically based on the assumption that the data is fairly normally distributed. In the health care domain it is common to find skewed data, for instance in surgical procedure times or pulse oxymetry [8]. Refer to Chap. 15-Exploratory Data Analysis for a formal definition of skewness. If a variable follows a log-normal distribution then the logarithms of the observations follow a normal distribution. A reasonable approach then is to apply the *ln* to the original data and they apply the tests intended to the "normalized" distributions. We refer to this method as the log-IQ.

14.3.5 Ordinary and Studentized Residuals

In a linear regression model, ordinary residuals are defined as the difference between the observed and predicted values. Data points with large residuals differ from the general regression trend and may represent outliers. The problem is that their magnitudes depend on their units of measurement, making it difficult to, for example, define a threshold at which a point is considered an outlier. Studentized residuals eliminate the units of measurement by dividing the residuals by an estimate of their standard deviation. One limitation of this approach is it assumes the regression model is correctly specified.

14.3.6 Cook's Distance

In a linear regression model, Cook's distance is used to estimate the influence of a data point on the regression. The principle of Cook's distance is to measure the effect of deleting a given observation. Data points with a large distance may represent outliers. For the *i*th point in the sample, Cook's distance is defined as:

$$D_i = \frac{\sum_{j=1}^{n}\left(\hat{y}_j\hat{y}_{j(i)}\right)^2}{(k+1)s^2} \tag{14.4}$$

Where $\hat{y}_{j(i)}$ is the prediction of y_j by the revised regression model when the *i*th point is removed from the sample, and s is the estimated root mean square error. Instinctively, D_i is a normalized measure of the influence of the point i on all predicted mean values $\hat{y}_j$ with $j = 1, \ldots, n$. Different cut-off values can be used for flagging highly influential points. Cook has suggested that a distance >1 represents a simple operational guideline [9]. Others have suggested a threshold of $4/n$, with n representing the number of observations.

14.3.7 Mahalanobis Distance

This test is based on Wilks method designed to detect a single outlier from a normal multivariable sample. It approaches the maximum squared Mahalanobis Distance (MD) to an F-distribution function formulation, which is often more appropriate than a χ^2 distribution [10]. For a p-dimensional multivariate sample x_i ($i = 1,\ldots,n$), the Mahalanobis distance of the *i*th case is defined as:

$$MD_i = \sqrt{(x_i - t)^T C^{-1}(x_i - t)} \tag{14.5}$$

where t is the estimated multivariate location, which is usually the arithmetic mean, and C is the estimated covariance matrix, usually the sample covariance matrix.

Multivariate outliers can be simply defined as observations having a large squared Mahalanobis distance. In this work, the squared Mahalanobis distance is compared with quantiles of the F-distribution with p and $p - 1$ degrees of freedom. Critical values are calculated using Bonferroni bounds.

14.4 Proximity Based Models

Proximity-based techniques are simple to implement and unlike statistical models they make no prior assumptions about the data distribution model. They are suitable for both supervised and unsupervised multivariable outlier detection [4].

Clustering is a type of proximity-based technique that starts by partitioning a *N*–dimensional dataset into c subgroups of samples (clusters) based on their similarity. Then, some measure of the fit of the data points to the different clusters is used in order to determine if the data points are outliers [2]. One challenge associated with this type of technique is that it assumes specific shapes of clusters depending on the distance function used within the clustering algorithm. For example, in a 3-dimensional space, the Euclidean distance would consider spheres as equidistant, whereas the Mahalanobis distance would consider ellipsoids as equidistant (where the length of the ellipsoids in one axis is proportional to the variance of the data in that direction).

14.4.1 k-Means

The k-means algorithm is widely used in data mining due to its simplicity and scalability [11]. The difficulty associated with this algorithm is the need to determine *k*, the number of clusters, in advance. The algorithm minimizes the within-cluster sum of squares, the sum of distances between each point in a cluster and the cluster centroid. In k-means, the center of a group is the mean of measurements in the group. Metrics such as the Akaike Information Criterion or the Bayesian Information Criterion, which add a factor proportional to *k* to the cost function used during clustering, can help determine *k*. A *k* value which is too high will increase the cost function even if it reduces the within-cluster sum of squares [12, 13].

14.4.2 k-Medoids

Similarly to k-means, the k-medoids clustering algorithm partitions the dataset into groups so that it minimizes the sum of distances between a data point and its center. In contrast to the k-means algorithm, in k-medoids the cluster centers are members of the group. Consequently, if there is a region of outliers outside the area with higher density of points, the cluster center will not be pushed towards the outliers region, as in k-means. Thus, k-medoids is more robust towards outliers than k-means.

14.4.3 Criteria for Outlier Detection

After determining the position of the cluster center with either k-means or k-medoids, the criteria to classify an item as an outlier must be specified, and different options exist:

Criterion 1: The first criterion proposed to detect outliers is based on the Euclidean distance to the cluster centers *C*, such that points more distant to their center than the minimum interclusters distance are considered outliers:

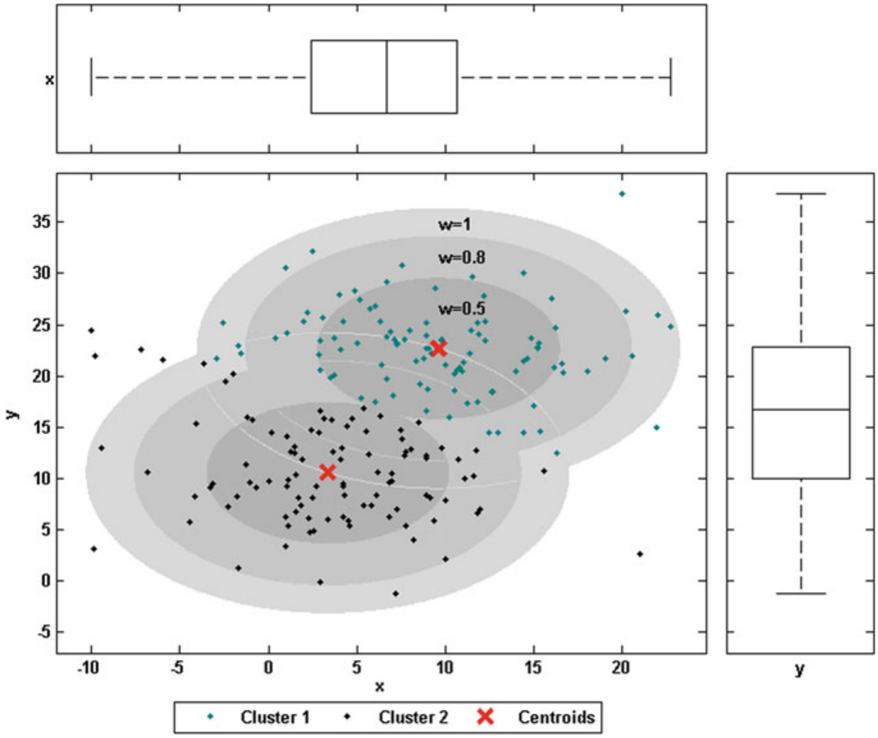


Fig. 14.2 Effect of different weights w in the detection of cluster-based outliers, using criterion 1

$$x \in C_k \textit{ is outlier if } d(x, C_k) > \min_{k \neq j}\{\delta(C_k, C_j)\} \times w \tag{14.6}$$

where $d(x, C_k)$ is the Euclidean distance between point x and C_k center, $\delta(C_k, C_j)$ is the distance between C_k and C_j centers and $w = \{0.5, 0.7, 1, 1.2, 1.5, \ldots\}$ is a weighting parameter that determines how aggressively the method will remove outliers.

Figure 14.2 provides a graphical example of the effect of varying values of w in the creation of boundaries for outlier detection. While small values of w aggressively remove outliers, as w increases the harder it is to identify them.

Criterion 2: In this criterion, we calculate the distance of each data point to its centroid (case of k-means) or medoid (case of k-medoids) [14]. If the ratio of the distance of the nearest point to the cluster center and these calculated distances are smaller than a certain threshold, than the point is considered an outlier. The threshold is defined by the user and should depend on the number of clusters selected, since the higher the number of clusters the closer are the points inside the cluster, i.e., the threshold should decrease with increasing c.

14.5 Supervised Outlier Detection

In many scenarios, previous knowledge about outliers may be available and can be used to label the data accordingly and to identify outliers of interest. The methods relying on previous examples of data outliers are referred to as supervised outlier detection methods, and involve training classification models which can later be used to identify outliers in the data. Supervised methods are often devised for anomaly detection in application domains where anomalies are considered occurrences of interest. Examples include fraud control, intrusion detection systems, web robot detection or medical diagnosis [1]. Hence, the labels represent what an analyst might be specifically looking for rather than what one might want to remove [2]. The key difference comparing to many other classification problems is the inherent unbalanced nature of data, since instances labeled as "abnormal" are present much less frequently than "normal" labeled instances. Interested readers can find further information about this topic in the textbook by Aggarwal, for instance [2].

14.6 Outlier Analysis Using Expert Knowledge

In univariate analyses, expert knowledge can be used to define thresholds of values that are normal, critical (life-threatening) or impossible because they fall outside permissible ranges or have no physical meaning [15]. Negative measurements of heart rate or body temperatures are examples of impossible values. It is very important to check the dataset for these types of outliers, as they originated undoubtedly from human error or equipment malfunction, and should be deleted or corrected.

14.7 Case Study: Identification of Outliers in the Indwelling Arterial Catheter (IAC) Study

In this section, various methods will be applied to identify outliers in two "real world" clinical datasets used in a study that investigated the effect of inserting an indwelling arterial catheter (IAC) in patients with respiratory failure. Two datasets are used, and include patients that received an IAC (IAC group) and patients that did not (non-IAC). The code used to generate the analyses and the figures is available in the GitHub repository for this book.

Table 14.1 Normal, critical and impossible ranges for the selected variables, and maximum and minimum values present in the datasets

	Reference value			Analyzed data		
Variable	Normal range	Critical	Impossible	IAC	Non-IAC	Units
Age	–	–	<17 (adults)	15.2–99.1	15.2–97.5	Years
SOFA	–	–	<0 and >24	1–17	0–14	No units
WBC	3 9–10.7	≥ 100	<0	0.3–86.0	0 2–109.8	$\times 10^9$ cells/L
Hemoglobin	Male: 13.5–17.5	≤ 6 and ≥ 20	<0	Male: 3 2–19.0	4.9–18.6	g/dL
	Female: 12–16			Female: 2.0–18.1	4.2–18.1	
Platelets	150–400	≤ 40 and ≥ 1000	<0	7.0–680.0	9.0–988.0	$\times 10^9$/L
Sodium	136–145	≤ 120 and ≥ 160	<0	105 0–165.0	111.0–154.0	mmol/L
Potassium	3.5–5	≤ 2.5 and ≥ 6	<0	1 9–9.8	1.9–8.3	mmol/L
TCO_2	22–28	≤ 10 and ≥ 40 [4]	<0	2.0–62.0	5.0–52.0	mmol/L
Chloride [29]	95–105	≤ 70 and ≥ 120	<0 and ≥ 160	81.0–133.0	78.0–127.0	mmol/L
BUN	7–18	≥ 100 [1]	<0	2.0–139.0	2.0-126.0	mg/dL
Creatinine	0.6–1.2	≥ 10	<0	0.2–12 5	0.0–18.3	mg/dL
PO_2	75–105	≤ 40	<0	25 0–594.0	22.0–634.0	mmHg
PCO_2	33–45	≤ 20 and ≥ 70	<0	8.0–141.0	14.0–158.0	mmHg

14.8 Expert Knowledge Analysis

Table 14.1 provides maximum and minimum values for defining normal, critical and permissible ranges in some of the variables analyzed in the study, as well as maximum and minimum values present in the dataset.

14.9 Univariate Analysis

In this section, univariate outliers are identified for each variable within pre-defined classes (survivors and non-survivors), using the statistical methods described above.

Table 14.2 summarizes the number and percentage of outliers identified by each method in the Indwelling Arterial Catheter (IAC) and non-IAC groups. Overall, Tukey's and log-IQ are the most conservative methods, i.e., they identify the

Table 14.2 Number and percentage of outliers identified by each method

	IAC									
	Class 0 (811 patients)					Class 1 (163 patients)				
	IQ	Tukey's	log–IQ	Z-score	Mod z-score	IQ	Tukey's	Log–IQ	Z-score	Mod z-score
Age	0 (0.0 %)	0 (0.0 %)	1 (0.1 %)	0 (0.0 %)	0 (0.0 %)	5 (0.6 %)	0 (0.0 %)	8 (1.0 %)	4 (0.5 %)	5 (0.6 %)
SOFA	13 (1.6 %)	0 (0.0 %)	6 (0.7 %)	2 (0.2 %)	20 (2.5 %)	**16 (2.0 %)**	3 (0.4 %)	8 (1.0 %)	1 (0.1 %)	5 (0.6 %)
WBC	20 (2.5 %)	3 (0.4 %)	21 (2.6 %)	5 (0.6 %)	10 (1.2 %)	6 (0.7 %)	1 (0.1 %)	5 (0.6 %)	1 (0.1 %)	3 (0.4 %)
Hemoglobin	8 (1.0 %)	1 (0.1 %)	13 (1.6 %)	5 (0.6 %)	4 (0.5 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)
Platelets	17 (2.1 %)	1 (0.1 %)	36 (4.4 %)	7 (0.9 %)	7 (0.9 %)	4 (0.5 %)	0 (0.0 %)	2 (0.2 %)	2 (0.2 %)	1 (0.1 %)
Sodium	30 (3.7 %)	8 (1.0 %)	30 (3.7 %)	10 (1.2 %)	26 (3.2 %)	8 (1.0 %)	1 (0.1 %)	8 (1.0 %)	2 (0.2 %)	2 (0.2 %)
Potassium	39 (4.8 %)	10 (1.2 %)	35 (4.3 %)	14 (1.7 %)	26 (3.2 %)	9 (1.1 %)	1 (0.1 %)	7 (0.9 %)	2 (0.2 %)	8 (1.0 %)
TCO_2	24 (3.0 %)	4 (0.5 %}	31 (3.8 %)	13 (1.6 %)	13 (1.6 %}	9 (1.1 %)	2 (0.2 %)	6 (0.7 %)	2 (0.2 %)	2 (0.2 %)
Chloride	21 (2.6 %)	3 (0.4 %)	24 (3.0 %)	13 (1.6 %)	18 (2.2 %)	4 (0.5 %)	0 (0.0 %)	3 (0.4 %)	1 (0.1 %)	1 (0.1 %)
BUN	**72 (8.9 %)**	**37 (4.6 %)**	**48 (5.9 %)**	**20 (2.5 %)**	**60 (7.4 %)**	13 (1.6 %)	**9 (1.1 %)**	7 (0.9 %)	**5 (0.6 %)**	**13 (1.6 %)**
Creatinine	50 (6.2 %)	31 (3.8 %)	43 (5.3 %)	18 (2.2 %)	40 (4.9 %)	11 (1.4 %)	2 (0.2 %)	2 (0.2 %)	2 (0.2 %)	8 (1.0 %)
PO_2	0 (0.0 %)	0 (0.0 %)	2 (0.2 %)	0 (0.0 %}	0 (0.0 %)	0 (0.0 %)	0(0.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)
PCO_2	53 (6.5 %)	22 (2.7 %)	**48 (5.9 %)**	19 (2.3 %}	37 (4.6 %)	11 (1.4 %)	4 (0.5 %)	**13 (1.6 %)**	4 (0.5 %)	9 (1.1 %)
Total patients	**220 (27.1 %)**	86(10.6 %)	210 (25.9 %)	91 (11.2 %)	165 (20.3 %)	**63 (7.8 %)**	20 (2.5 %)	47 (5.8 %)	23 (2.8 %)	43 (5.3 %)
	Non-IAC									
	Class 0 (524 patients)					Class 1 (83 patients)				
	IQ	Tukey's	log–IQ	Z-score	Mod z-score	IQ	Tukey's	Log–IQ	Z-score	Mod z-score
Age	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %>	0 (0.0 %)	1 (0.2 %)	0 (0.0 %)	3 (0.6 %)	1 (0.2 %)	1 (0.2 %)
SOFA	**51 (9.7 %)**	2 (0.4 %)	**48 (9.2 %)**	2 (0.4 %)	7 (1.3 %)	**9 (1.7 %)**	1 (0.2 %)	8 (1.5 %)	1 (0.2 %)	3 (0.6 %)
WBC	21 (4.0 %)	4 (0.8 %}	10 (1.9 %)	4 (0.11 %)	11 (2.1 %)	4 (0.8 %)	1 (0.2 %)	4 (0.8 %)	1 (0.2 %)	3 (0.6 %)
Hemoglobin	1 (0.4 %|	0 (0.0 %)	6 (1.1 %)	2 (0.4 %)	2 (0.4 %)	0 (0.0 %)	0 (0.0 %)	2 (0.4 %)	0 (0.0 %)	0 (0.0 %)

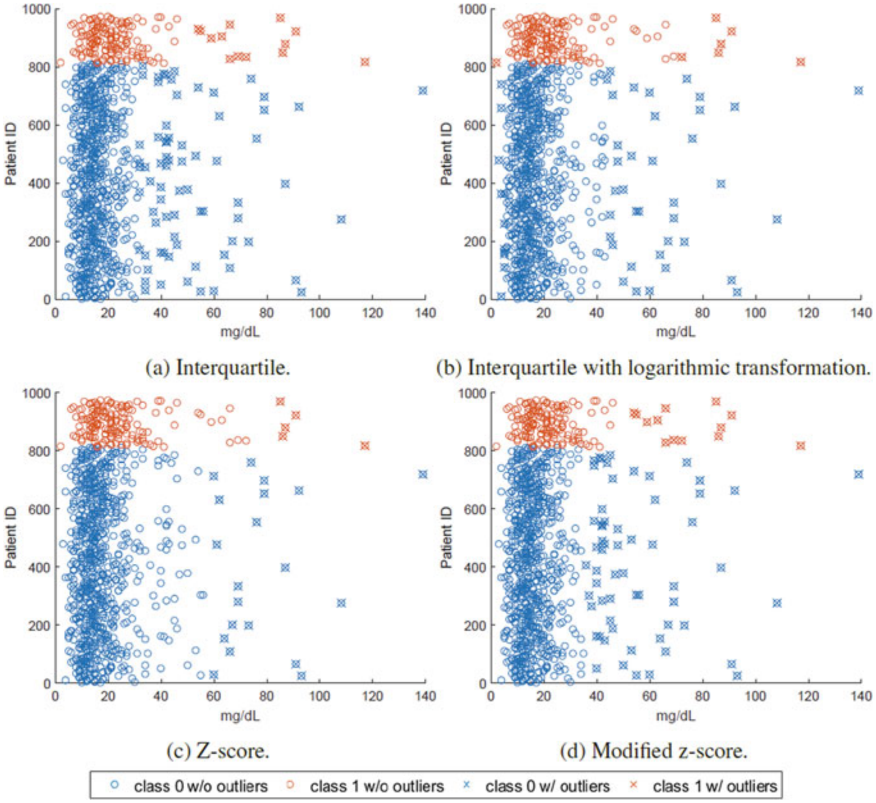
(continued)

Table 14.2 (continued)

	Non-IAC									
	Class 0 (524 patients)					Class 1 (83 patients)				
	IQ	Tukey's	log–IQ	Z-score	Mod z-score	IQ	Tukey's	Log–IQ	Z-score	Mod z-score
Platelets	15 (2.9 %)	5 (1.0 %)	21 (4.0 %)	5 (1.0 %)	6 (1.1 %)	4 (0.8 %)	1 (0.2 %)	5 (1.0 %)	**2 (0.4 %)**	2 (0.4 %)
Sodium	25 (4.8 %)	9 (1.7 %)	25 (4.11 %)	9 (1.7 %)	20 (3.11 %)	5 (1.0 %)	1 (0.2 %)	5 (1.0 %)	1 (0.2 %)	1 (0.2 %)
Potassium	22 (4.2 %)	2 (0.4 %)	14 (2.7 %)	6 (1.1 %)	14 (2.7 %)	1 (0.2 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)
TCO_2	27 (5.2 %)	4 (0.8 %)	31 (5.9 %)	8 (1.5 %)	5 (1.0 %}	4 (0.8 %)	1 (0.2 %)	4 (0.8 %)	**2 (0.4 %)**	3 (0.6 %)
Chloride	21 (4.0 %)	4 (0.8 %)	20 (3.11 %)	9 (1.7 %)	11 (2.1 %)	**9 (1.7 %)**	1 (0.2 %)	**9 (1.7 %)**	1 (0.2 %)	4 (0.8 %)
BUN	35 (6.7 %)	**20 (3.8 %)**	27 (5.2 %)	**13 (2.5 %)**	**34 (6.5 %)**	6 (1.1 %)	2 (0.4 %)	2 (0.4 %)	**2 (0.4 %)**	6 (1.1 %)
Creatinine	29 (5.5 %)	17 (3.2 %)	25 (4.8 %)	8 (1.5 %)	22 (4.2 %)	7 (1.3 %)	2 (0.4 %)	3 (0.6 %)	**2 (0.4 %)**	5 (1.0 %)
PO_2	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	1 (0.2 %)	0 (0.0 %)	0 (0.0 %)	0 (0.0 %)	3 (0.6 %)
PCO_2	34 (6.5 %)	11 (2.1 %)	33 (6.3 %)	10 (1.9 %)	28 (5.3 %)	8 (1.5 %)	**4 (0.8 %)**	6 (1.1 %)	**2 (0.4 %)**	**8 (1.5 %)**
Total patients	**176 (33.6 %)**	59 (11.3 %)	172 (32.8 %)	56 (10.7 %)	111 (21.2 %}	**37 (7.1 %)**	11 (2.1 %)	29 (5.5 %)	11 (2.1 %)	28 (5.3 %)

"Total patients" represents the number of patients identified when considering all variables together. The results in bold highlight the variable with the most outliers in each method, and also the method that removes more patients in total, in each class. Class 0: represents survivors, Class 1: non-survivors smallest number of points as outliers, whereas IQ identifies more outliers than any other method. With a few exceptions, the modified z-score identifies more outliers than the z-score.

A preliminary investigation of results showed that values falling within reference normal ranges (see Table 14.1) are never identified as outliers, whatever the method. On the other hand, critical values are often identified as such. Additional remarks can be made as in general (1) more outliers are identified in the variable BUN than in any other and (2) the ratio of number of outliers and total number of patients is smaller in the class 1 cohorts (non-survivors). As expected, for variables that approximate more to lognormal distribution than to a normal distribution, such as potassium, BUN and PCO2, the IQ method applied to the logarithmic transformation of data (log-IQ method) identifies less outliers than the IQ applied to the real data. Consider for instance the variable BUN, which follows approximately a lognormal distribution. Figure 14.3 shows a scatter of all data points and the identified outliers in the IAC group.



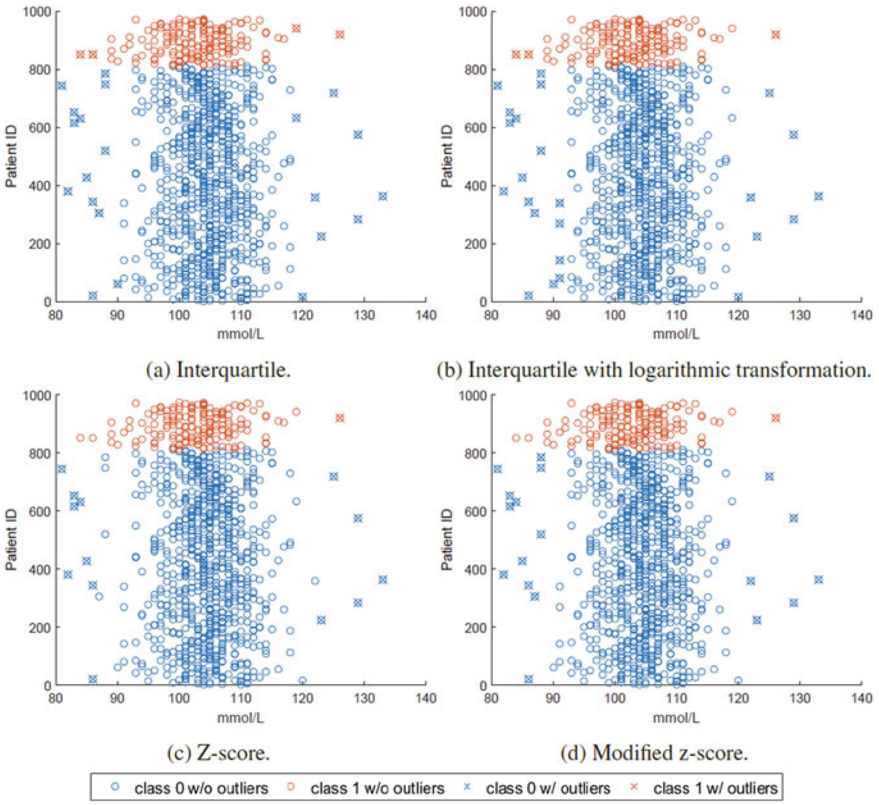
(a) Interquartile.

(b) Interquartile with logarithmic transformation.

(c) Z-score.

(d) Modified z-score.

Fig. 14.3 Outliers identified by statistical analysis for the variable BUN, in the IAC cohort. Class 0: survivors; Class 1: non survivors



(a) Interquartile.

(b) Interquartile with logarithmic transformation.

(c) Z-score.

(d) Modified z-score.

Fig. 14.4 Outliers identified by statistical analysis for the variable chloride, in the IAC cohort. Class 0: survivors; Class 1: non survivors

On the other hand, when the values follow approximately a normal distribution, as in the case of chloride (see Fig. 14.4), the IQ method identifies less outliers than log-IQ. Of note, the range of values considered outliers differs between classes, i.e., what is considered an outlier in class 0 is not necessarily an outlier in class 1. An example of this is values smaller than 90 mmol/L in the modified z-score.

Since this is a univariate analysis, the investigation of extreme values using expert knowledge is of interest. For chloride, normal values are in the range of 95–105 mmol/L, whereas values <70 or >120 mmol/L are considered critical, and concentrations above 160 mmol/L are physiologically impossible [15]. Figure 14.4 confirms that normal values are always kept, whatever the method. Importantly, some critical values are not identified in both z-score and modified z-score (especially in class 1). Thus, it seems that the methods identify outliers that should not be eliminated, as they likely represent actual values in extremely sick patients.

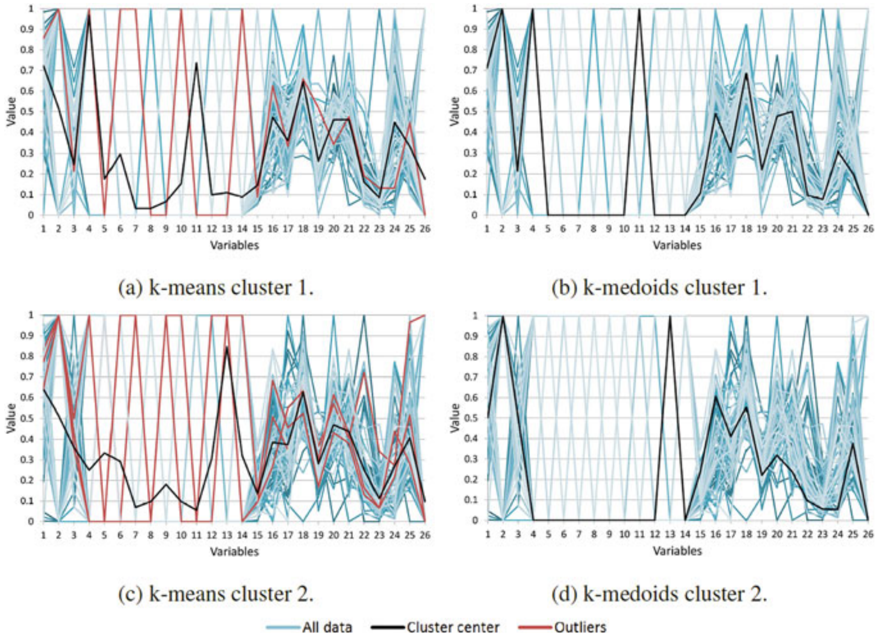
14.10 Multivariable Analysis

Using model based approaches, unusual combination of values for a number of variables can be identified. In this analysis we will be concerned with multivariable outliers for the complete set of variables in the datasets, including those that are binary. In order to investigate multivariable outliers in IAC and non-IAC patients, the Mahalanobis distance and cluster based approaches are tested within pre-defined classes. Table 14.3 shows the average results in terms of number of clusters *c* determined by the silhouette index, and the percentage of patients identified as

Table 14.3 Multivariable outliers identified by k-means, k-medoids and Mahalanobis distance

	Criterion	Weight	*c*		% of outliers Class 0	
			Class 0	Class 1	Class 0	Class 1
IAC						
K-means, silhouette index	1	1.2	4 ± 3.1	2 ± 0.0	25.2 ± 7.4	20.9 ± 11.0
	1	1.5	3 ± 2.9	2 ± 0.0	7.9 ± 4.6	3.3 ± 5.9
	1	1.7	3 ± 2.6	2 ± 0.0	3.6 ± 2.5	0.4 ± 2.2
	1	2.0	4 ± 3.1	2 ± 0.0	1.0 ± 1.1	0.1 ± 0.3
K-means, *c* = 2	2	0.05	2 ± 0.0	2 ± 0.0	28.5 ± 4.8	21.4 ± 11.9
	2	0.06	2 ± 0.0	2 ± 0.0	9.3 ± 4.2	2.9 ± 5.2
K-medoids, silhouette index	1	1.2	4 ± 3.0	2 ± 0.0	4.1 ± 2.2	0.8 ± 3.1
	1	1.5	3 ± 2.6	2 ± 0.0	1.1 ± 1.0	0.1 ± 0.3
	1	1.7	3 ± 2.9	2 ± 0.0	0.2 ± 0.2	0.0 ± 0.0
	1	2.0	4 ± 3.0	2 ± 0.0	0.7 ± 0.4	0.0 ± 0.0
K-medoids, *c* = 2	2	0.01	2 ± 0.0	2 ± 0.0	34.6 ± 8.6	2.5 ± 0.0
	2	0.02	2 ± 0.0	2 ± 0.0	20.8 ± 6.1	0.0 ± 0.0
Mahalanobis	–	–	–	–	16.7 ± 5.5	0.0 ± 0.0
Non-IAC						
K-means, silhouette index	1	1.2	9 ± 1.8	7 ± 2.4	12.8 ± 4.1	13.0 ± 9.5
	1	1.5	9 ± 1.7	7 ± 2.5	2.8 ± 1.8	1.0 ± 1.7
	1	1.7	9 ± 1.8	7 ± 2.5	0.9 ± 1.2	0.0 ± 0.2
	1	2.0	9 ± 2.4	7 ± 2.5	0.2 ± 0.7	0.0 ± 0.0
K-means, *c* = 2	2	0.05	2 ± 0.0	2 ± 0.0	25.5 ± 4.5	41.0 ± 11.9
	2	0.06	2 ± 0.0	2 ± 0.0	10.6 ± 2.6	4.8 ± 7.2
K-medoids, silhouette index	1	1.2	9 ± 1.5	7 ± 2.5	3.8 ± 1.6	1.4 ± 1.6
	1	1.5	9 ± 2.0	7 ± 2.4	0.9 ± 1.9	0.0 ± 0.0
	1	1.7	9 ± 2.0	7 ± 2.4	0.3 ± 0.6	0.0 ± 0.0
	1	2.0	9 ± 1.3	7 ± 2.5	0.4 ± 0.9	0.0 ± 0.0
K-medoids, *c* = 2	2	0.01	2 ± 0.0	2 ± 0.0	19.7 ± 4.0	2.7 ± 8.8
	2	0.02	2 ± 0.0	2 ± 0.0	11.0 ± 2.8	1.0 ± 5.0
Mahalanobis	–	–	–	–	6.8 ± 2.6	0.8 ± 4.0

Results are presented as mean ± standard deviation



(a) k-means cluster 1.

(b) k-medoids cluster 1.

(c) k-means cluster 2.

(d) k-medoids cluster 2.

Fig. 14.5 Outliers identified by clustering based approaches for patients that died after IAC. Criterion 1, based on interclusters distance, with $c = 2$ and $w = 1.5$ was used. K-medoids does not identify outliers, whereas k-means identifies 1 outlier in cluster 1 and 2 outliers in cluster 2

outliers. In order to account for variability, the tests were performed 100 times. The data was normalized for testing the cluster based approaches only.

Considering the scenario where two clusters are created for the complete IAC dataset separated by classes, we investigate outliers by looking at multivariable observations around cluster centers. Figure 14.5 shows an example of the outliers detected using k-means and k-medoids with the criterion 1 and weight equal to 1.5. For illustrative purposes, we present only the graphical results of patients that died in the IAC group (class 1). The x-axis represents each of the selected features (see Table 14.1) and the y-axis represents the corresponding values normalized between 0 and 1. K-medoids does not identify any outlier, whereas k-means identifies 1 outlier in the first cluster and 2 outliers in the second cluster. This difference can be attributed to the fact that the intercluster distance is smaller in k-medoids than in k-means.

The detection of outliers seems to be more influenced by binary features than by continuous features: red lines are, with some exceptions, fairly close to black lines for the continuous variables (1 to 2 and 15 to 25) and distant in the binary variables. A possible explanation is that clustering was essentially designed for multivariable continuous data; binary variables produce a maximum separation, since only two values exist, 0 and 1, with nothing between them.

14.11 Classification of Mortality in IAC and Non-IAC Patients

Logistic regression models were created to assess the effect of removing outliers using the different methods in the classification of mortality in IAC and non-IAC patients, following the same rationale as in Chap. 13-Missing Data. A 10-fold cross validation approach was used to assess the validity and robustness of the models. In each round, every outlier identification method was applied separately for each class of the training set, and the results were averaged over the rounds. Before cross-validation, the values were normalized between 0 and 1 using the min-max procedure. For the log-IQ method, the data was log-transformed before normalization, except for variables containing null values (binary variables in Table 14.1, SOFA and creatinine). We also investigate the scenario where only the 10 % worst examples detected by each statistical method within each class are considered, and the case where no outliers were removed (all data is used). In the clustering based approaches, the number of clusters *c* was chosen between 2 and 10 using the silhouette index method. We also show the case where *c* is fixed as 2. The weight of the clustering based approaches was adjusted according to the particularities of the method. Since a cluster center in k-medoids is a data point belonging to the dataset, the distance to its nearest neighbor is smaller than in the case of k-means, especially because a lot of binary variables are considered. For this reason, we chose higher values of *w* for k-means criterion 2.

The performance of the models is evaluated in terms of area under the receiver operating characteristic curve (AUC), accuracy (ACC, correct classification rate), sensitivity (true positive classification rate), and specificity (true negative classification rate). A specific test suggested by DeLong and DeLong can then test whether the results differ significantly [16].

The performance results for the IAC group are shown in Table 14.4, and the percentage of patients removed using each method in Table 14.5. For conciseness, the results for the non-IAC group are not shown. The best performance for IAC is AUC = 0.83 and ACC = 0.78 (highlighted in bold). The maximum sensitivity is 87 % and maximum specificity is 79 %, however these two do not occur simultaneously. Overall, the best AUC is obtained when all the data is used and when only a few outliers are removed. The worst performances are obtained using the z-score without trimming the results and k-means and k-medoids using $c = 2$, criterion 1 and weight 1.2. As for non-IAC, the best performance corresponds to AUC = 0.88, ACC = 0.84, sensitivity = 0.85 and specificity = 0.85. Again, the best performance is achieved when all the data is used and in the cases where less outliers are removed. The worst performance by far is obtained when all outliers identified by the z-score are removed. Similarly to IAC, for k-means and k-medoids criterion 1, increasing values of weight provide better results.

Table 14.4 IAC logistic regression results using 10-fold cross validation, after removal of outliers and using the original dataset

Statistical	Cutoff	AUC	ACC	Sensitivity	Specificity
IQ	–	0.81 ± 0.05	0.76 ± 0.05	0.71 ± 0.14	0.76 ± 0.06
	10	0.82 ± 0.06	0.77 ± 0.06	0.76 ± 0.11	0.77 ± 0.07
Tukey's	–	0.82 ± 0.05	0.75 ± 0.06	0.76 ± 0.09	0.75 ± 0.06
	10	**0.83 ± 0.06**	0 78 ± 0.05	0.75 ± 0 10	0.78 ± 0.06
Log-IQ	–	0.82 ± 0.06	0.76 ± 0.05	0.74 ± 0 14	0.76 ± 0.06
	10	**0.83 ± 0.06**	**0.78 ± 0.04**	0.73 ± 0 10	0.79 ± 0.05
Z-score	–	0.78 ± 0.03	0.67 ± 0.06	0.85 ± 0 09	0.64 ± 0.08
	10	0.81 ± 0.07	0.75 ± 0.06	0.74 ± 013	0.75 ± 0.07
Modified z-score	–	0.82 ± 0.05	0.76 ± 0.05	0.77 ± 0 14	0.76 ± 0.05
	10	0.82 ± 0.06	0.77 ± 0.06	0.75 ± 0 10	0.77 ± 0.06
Mahalanobis	–	0.81 ± 0.08	0.75 ± 0.06	0.73 ± 0 10	0.76 ± 0.07
Cluster based	Weight	AUC	ACC	Sensitivity	Specificity
K-means silhouette criterion 1	1.2	0.81 ± 0.08	0.72 ± 0.05	0.80 ± 0.12	0.70 ± 0.06
	1.5	0.82 ± 0.05	0.76 ± 0.06	0.76 ± 011	0.76 ± 0.06
	1.7	**0.83 ± 0.06**	**0.78 ± 0.05**	0.77 ± 0 10	0.78 ± 0.06
	2	**0.83 ± 0.06**	**0.78 ± 0.05**	0.74 ± 0.09	0.78 ± 0.06
K-means $c = 2$ criterion 1	1.2	0.79 ± 0.08	0.66 ± 0.05	0.84 ± 0 10	0.63 ± 0.06
	1.5	0.82 ± 0.06	0.73 ± 0.06	0.79 ± 0 09	0.72 ± 0.07
	1.7	0.82 ± 0.06	0.75 ± 0.06	0.78 ± 0.08	0.75 ± 0.08
	2	**0.83 ± 0.07**	**0.78 ± 0.06**	0.76 ± 0 09	0.78 ± 0.06
K-means criterion 2	0 05	**0.83 ± 0.07**	0.77 ± 0.05	0.74 ± 0.09	0.78 ± 0.06
	0.06	**0.83 ± 0.06**	0.77 ± 0.06	0.75 ± 0 10	0.78 ± 0.06
K-medoids silhouette criterion 1	1.2	0.81 ± 0.04	0.68 ± 0.04	0.85 ± 0 09	0.64 ± 0.05
	1.5	**0.83 ± 0.05**	0.74 ± 0.04	0.80 ± 0 10	0.73 ± 0.06
	1.7	**0.83** ± **0.05**	0.75 ± 0.06	0.78 ± 0 10	0.74 ± 0.07
	2	**0.83** ± 0.**06**	0.77 ± 0.05	0.77 ± 0 09	0.77 ± 0.06
K-medoids $c = 2$ criterion 1	1.2	0.78 ± 0.06	0.62 ± 0.07	0.87 ± 0 08	0.57 ± 0.07
	1.5	0.81 ± 0.06	0.70 ± 0.06	0.83 ± 0 10	0.68 ± 0.08
	1.7	0.82 ± 0.06	0.72 ± 0.06	0.80 ± 0 10	0.71 ± 0.08
	2	**0.83 ± 0.07**	0.76 ± 0.06	0.77 ± 0 10	0.75 ± 0.07
K-medoids criterion 2	0.01	**0.83** ± 0.**07**	0.74 ± 0.07	0.77 ± 0 10	0.74 ± 0.08
	0 02	0.81 ± 0.06	0.67 ± 0.06	0.85 ± 0 09	0.63 ± 0.08
All data	–	**0.83 ± 0.06**	**0.78 ± 0.05**	0.76 ± 0.11	0.79 ± 0.06

Results are presented as mean ± standard deviation

Table 14.5 Percentage of IAC patients removed by each method in the train set, during cross-validation

Statistical	Cutoff	Class 0	Class 1	Total
IQ	–	23.1 ± 1.4	33.3 ± 1.9	24.8 ± 1.4
	10	3.3 ± 0.2	5.2 ± 0.3	3.6 ± 0.2
Tukey's	–	8.7 ± 0.05	10.1 ± 1.1	9.0 ± 0.5
	10	1.2 ± 0.1	1.3 ± 0.2	1.3 ± 0 1
Log-IQ	–	22.8 ± 1.1	25.4 ± 2.0	23.2 ± 1.1
	10	3.1 ± 0.2	3.7 ± 0.5	3.2 ± 0 1
Z-score	–	35.0 ± 1.6	0.67 ± 0.06	32.6 ± 1.4
	10	5.3 ± 0.2	2.9 ± 1.3	4.9 ± 0.3
Modified z-score	–	18.3 ± 0.05	24.5 ± 1.3	19.4 ± 0.5
	10	2.4 ± 0.1	3.5 ± 0.4	2.6 ± 0.1
Mahalanobis	–	19.6 ± 9.6	17.4 ± 3.0	19.2 ± 8.1

Cluster based	Weight	Class 0	Class 1	Total
K-means silhouette criterion 1	1.2	19.6 ± 9.6	17.4 ± 3.0	19.2 ± 8.1
	1.5	6.1 ± 5.1	1.9 ± 0.5	5.4 ± 4.2
	1.7	2.5 ± 2.6	0.3 ± 0.3	2.2 ± 2.2
	2	0.7 ± 0.9	0.0 ± 0.0	0.6 ± 0.8
K-means c = 2 criterion 1	1.2	29.7 ± 3.5	17.4 ± 3.0	27.6 ± 2.9
	1.5	11.9 ± 3.0	1.9 ± 0.5	10.2 ± 2.5
	1.7	5.5 ± 2.0	0.3 ± 0.3	4.7 ± 1.6
	2	1.7 ± 0.8	0.0 ± 0.0	1.4 ± 0 7
K-means criterion 2	0 05	0.3 ± 0.2	0.0 ± 0.0	0.3 ± 0.2
	0.06	1.1 ± 0.5	0.0 ± 0.0	0.9 ± 0 4
K-medoids silhouette criterion 1	1.2	25.0 ± 10.7	3.8 ± 2.0	21.5 ± 8.8
	1.5	12.9 ± 7.4	0.0 ± 0.0	10.8 ± 6.2
	1.7	9.5 ± 6.1	0.0 ± 0.0	7.9 ± 5.1
	2	3.1 ± 2.3	0.0 ± 0.0	2.5 ± 1.9
K-medoids c = 2 criterion 1	1.2	34.7 ± 0.7	3.8 ± 2.0	29.5 ± 0.7
	1.5	19.6 ± 0.6	0.0 ± 0.0	16.3 ± 0 5
	1.7	14.9 ± 1.1	0.0 ± 0.0	12.4 ± 0 9
	2	5.1 ± 0.4	0.0 ± 0.0	4.2 ± 0 4
K-medoids criterion 2	0.01	8.3 ± 2.1	0.0 ± 0.0	6.9 ± 1.7
	0 02	28.9 ± 3.9	1.8 ± 3.8	24.4 ± 3.6

Results are presented as mean ± standard deviation

14.12 Conclusions and Summary

The univariable outlier analysis provided in the case study showed that a large number of outliers were identified for each variable within the predefined classes, meaning that the removal of all the identified outliers would cause a large portion of data to be excluded. For this reason, ranking the univariate outliers according to score values and discarding only those with highest scores provided better classification results.

Overall, none of the outlier removal techniques was able to improve the performance of a classification model. As it had been cleaned these results suggest that the dataset did not contain impossible values, extreme values are probably due to biological variation rather than experimental mistakes. Hence, the "outliers" in this study appear to contain useful information in their extreme values, and automatically excluding resulted in a loss of this information.

Some modeling methods already accommodate for outliers so they have minimal impact in the model, and can be tuned to be more or less sensitive to them. Thus, rather than excluding outliers from the dataset before the modeling step, an alternative strategy would be to use models that are robust to outliers, such as robust regression.

Take Home Messages

1. Distinguishing outliers as useful or uninformative is not clear cut.
2. In certain contexts, outliers may represent extremely valuable information that must not be discarded.
3. Various methods exist and will identify possible or likely outliers, but the expert eye must prevail before deleting or correcting outliers.

Code Appendix

The code used in this chapter is available in the GitHub repository for this book: https://github.com/MIT-LCP/critical-data-book. Further information on the code is available from this website.

References

1. Barnett V, Lewis T (1994) Outliers in statistical data, 3rd edn. Wiley, Chichester
2. Aggarwal CC (2013) Outlier analysis. Springer, New York
3. Osborne JW, Overbay A (2004) The power of outliers (and why researchers should always check for them). Pract Assess Res Eval 9(6):1–12
4. Hodge VJ, Austin J (2004) A survey of outlier detection methodologies. Artif Intell Rev 22 (2):85–126
5. Tukey J (1977) Exploratory data analysis. Pearson
6. Shiffler RE (1988) Maximum Z scores and outliers. Am Stat 42(1):79–80
7. Iglewicz B, Hoaglin DC (1993) How to detect and handle outliers. ASQC Quality Press
8. Seo S (2006) A review and comparison of methods for detecting outliers in univariate data sets. 09 Aug 2006 [Online]. Available: http://d-scholarship.pitt.edu/7948/. Accessed 07-Feb-2016
9. Cook RD, Weisberg S (1982) Residuals and influence in regression. Chapman and Hall, New York
10. Penny KI (1996) Appropriate critical values when testing for a single multivariate outlier by using the Mahalanobis distance. Appl Stat 45(1):73–81
11. Macqueen J (1967) Some methods for classification and analysis of multivariate observations. Presented at the proceedings of 5th Berkeley symposium on mathematical statistics and probability, pp 281–297
12. Hu X, Xu L (2003) A comparative study of several cluster number selection criteria. In: Liu J, Cheung Y, Yin H (eds) Intelligent data engineering and automated learning. Springer, Berlin, pp 195–202
13. Jones RH (2011) Bayesian information criterion for longitudinal and clustered data. Stat Med 30(25):3050–3056
14. Cherednichenko S (2005) Outlier detection in clustering
15. Provan D (2010) Oxford handbook of clinical and laboratory investigation. OUP Oxford
16. DeLong ER, DeLong DM, Clarke-Pearson DL (1988) Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. Biometrics 44 (3):837–845

Chapter 15
Exploratory Data Analysis

Matthieu Komorowski, Dominic C. Marshall, Justin D. Salciccioli and Yves Crutain

Learning Objectives

- Why is EDA important during the initial exploration of a dataset?
- What are the most essential tools of graphical and non-graphical EDA?

15.1 Introduction

Exploratory data analysis (EDA) is an essential step in any research analysis. The primary aim with exploratory analysis is to examine the data for distribution, outliers and anomalies to direct specific testing of your hypothesis. It also provides tools for hypothesis generation by visualizing and understanding the data usually through graphical representation [1]. EDA aims to assist the natural patterns recognition of the analyst. Finally, feature selection techniques often fall into EDA. Since the seminal work of Tukey in 1977, EDA has gained a large following as the gold standard methodology to analyze a data set [2, 3]. According to Howard Seltman (Carnegie Mellon University), "loosely speaking, any method of looking at data that does not include formal statistical modeling and inference falls under the term exploratory data analysis" [4].

EDA is a fundamental early step after data collection (see Chap. 11) and pre-processing (see Chap. 12), where the data is simply visualized, plotted, manipulated, without any assumptions, in order to help assessing the quality of the data and building models. "Most EDA techniques are graphical in nature with a few quantitative techniques. The reason for the heavy reliance on graphics is that by its very nature the main role of EDA is to explore, and graphics gives the analysts unparalleled power to do so, while being ready to gain insight into the data. There are many ways to categorize the many EDA techniques" [5].

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-43742-2_15) contains supplementary material, which is available to authorized users.

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_15

The interested reader will find further information in the textbooks of Hill and Lewicki [6] or the NIST/SEMATECH e-Handbook [1]. Relevant R packages are available on the CRAN website [7].

The objectives of EDA can be summarized as follows:

1. Maximize insight into the database/understand the database structure;
2. Visualize potential relationships (direction and magnitude) between exposure and outcome variables;
3. Detect outliers and anomalies (values that are significantly different from the other observations);
4. Develop parsimonious models (a predictive or explanatory model that performs with as few exposure variables as possible) or preliminary selection of appropriate models;
5. Extract and create clinically relevant variables.

EDA methods can be cross-classified as:

- Graphical or non-graphical methods
- Univariate (only one variable, exposure or outcome) or multivariate (several exposure variables alone or with an outcome variable) methods.

15.2 Part 1—Theoretical Concepts

15.2.1 Suggested EDA Techniques

Tables 15.1 and 15.2 suggest a few EDA techniques depending on the type of data and the objective of the analysis.

Table 15.1 Suggested EDA techniques depending on the type of data

Type of data	Suggested EDA techniques
Categorical	Descriptive statistics
Univariate continuous	Line plot, Histograms
Bivariate continuous	2D scatter plots
2D arrays	Heatmap
Multivariate: trivariate	3D scatter plot or 2D scatter plot with a 3rd variable represented in different color, shape or size
Multiple groups	Side-by-side boxplot

Table 15.2 Most useful EDA techniques depending on the objective

Objective	Suggested EDA techniques
Getting an idea of the distribution of a variable	Histogram
Finding outliers	Histogram, scatterplots, box-and-whisker plots
Quantify the relationship between two variables (one exposure and one outcome)	2D scatter plot +/curve fitting Covariance and correlation
Visualize the relationship between two exposure variables and one outcome variable	Heatmap
Visualization of high-dimensional data	t-SNE or PCA + 2D/3D scatterplot

t-SNE t-distributed stochastic neighbor embedding, *PCA* Principal component analysis

Table 15.3 Example of tabulation table

	Group count	Frequency (%)
Green ball	15	75
Red ball	5	25
Total	20	100

15.2.2 Non-graphical EDA

These non-graphical methods will provide insight into the characteristics and the distribution of the variable(s) of interest.

Univariate Non-graphical EDA

Tabulation of Categorical Data (Tabulation of the Frequency of Each Category)

A simple univariate non-graphical EDA method for categorical variables is to build a table containing the count and the fraction (or frequency) of data of each category. An example of tabulation is shown in the case study (Table 15.3).

Characteristics of Quantitative Data: Central Tendency, Spread, Shape of the Distribution (Skewness, Kurtosis)

Sample statistics express the characteristics of a sample using a limited set of parameters. They are generally seen as estimates of the corresponding population parameters from which the sample comes from. These characteristics can express the central tendency of the data (arithmetic mean, median, mode), its spread (variance, standard deviation, interquartile range, maximum and minimum value) or some features of its distribution (skewness, kurtosis). Many of those characteristics can easily be seen qualitatively on a histogram (see below). Note that these characteristics can only be used for quantitative variables (not categorical).

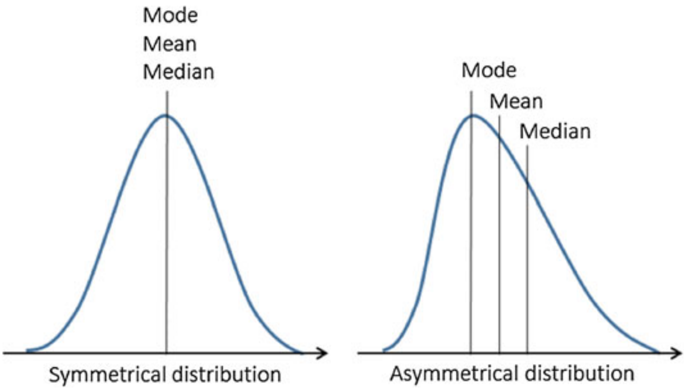


Fig. 15.1 Symmetrical versus asymmetrical (skewed) distribution, showing mode, mean and median

Central tendency parameters

The arithmetic mean, or simply called the mean is the sum of all data divided by the number of values. The median is the middle value in a list containing all the values sorted. Because the median is affected little by extreme values and outliers, it is said to be more "robust" than the mean (Fig. 15.1).

Variance

When calculated on the entirety of the data of a population (which rarely occurs), the variance σ^2 is obtained by dividing the sum of squares by n, the size of the population.

The sample formula for the variance of observed data conventionally has n-1 in the denominator instead of n to achieve the property of "unbiasedness", which roughly means that when calculated for many different random samples from the same population, the average should match the corresponding population quantity (here σ^2). s^2 is an unbiased estimator of the population variance σ^2.

$$s^2 = \frac{\sum_{i=1}^{n} (x_i - \underline{x})^2}{(n-1)} \tag{15.1}$$

The standard deviation is simply the square root of the variance. Therefore it has the same units as the original data, which helps make it more interpretable.

The sample standard deviation is usually represented by the symbol s. For a theoretical Gaussian distribution, mean plus or minus 1, 2 or 3 standard deviations holds 68.3, 95.4 and 99.7 % of the probability density, respectively.

Interquartile range (IQR)

The IQR is calculated using the boundaries of data situated between the 1st and the 3rd quartiles. Please refer to the Chap. 13 “Noise versus Outliers” for further detail about the IQR.

$$IQR = Q_3 - Q_1 \tag{15.2}$$

In the same way that the median is more robust than the mean, the IQR is a more robust measure of spread than variance and standard deviation and should therefore be preferred for small or asymmetrical distributions.

Important rule:

- **Symmetrical distribution** (not necessarily normal) **and N > 30**: express results as mean ± standard deviation.
- **Asymmetrical distribution or N < 30 or evidence for outliers:** use median ± IQR, which are more robust.

Skewness/kurtosis

Skewness is a measure of a distribution’s asymmetry. Kurtosis is a summary statistic communicating information about the tails (the smallest and largest values) of the distribution. Both quantities can be used as a means to communicate information about the distribution of the data when graphical methods cannot be used. More information about these quantities can be found in [9]).

Summary

We provide as a reference some of the common functions in R language for generating summary statistics relating to measures of central tendency (Table 15.4).

Testing the Distribution

Several non-graphical methods exist to assess the normality of a data set (whether it was sampled from a normal distribution), like the Shapiro-Wilk test for example. Please refer to the function called “Distribution” in the GitHub repository for this book (see code appendix at the end of this Chapter).

Table 15.4 Main R functions for basic measure of central tendencies and variability

Function	Description
summary(x)	General description of a vector
max(x)	Maximum value
mean(x)	Average or mean value
median(x)	Median value
min(x)	Smallest value
sd(x)	Standard deviation
var(x)	Variance, measure the spread or dispersion of the values
IQR(x)	Interquartile range

Finding Outliers

Several statistical methods for outlier detection fall into EDA techniques, like Tukey's method, Z-score, studentized residuals, etc [8]. Please refer to the Chap. 14 "Noise versus Outliers" for more detail about this topic.

Multivariate Non-graphical EDA

Cross-Tabulation

Cross-tabulation represents the basic bivariate non-graphical EDA technique. It is an extension of tabulation that works for categorical data and quantitative data with only a few variables. For two variables, build a two-way table with column headings matching the levels of one variable and row headings matching the levels of the other variable, then fill in the counts of all subjects that share a pair of levels. The two variables may be both exposure, both outcome variables, or one of each.

Covariance and Correlation

Covariance and correlation measure the degree of the relationship between two random variables and express how much they change together (Fig. 15.2).

The covariance is computed as follows:

$$cov(x, y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{n - 1} \tag{15.3}$$

where x and y are the variables, n the number of data points in the sample, $\bar{x}$ the mean of the variable x and $\bar{y}$ the mean of the variable y.

A positive covariance means the variables are positively related (they move together in the same direction), while a negative covariance means the variables are inversely related. A problem with covariance is that its value depends on the scale of the values of the random variables. The larger the values of x and y, the larger the

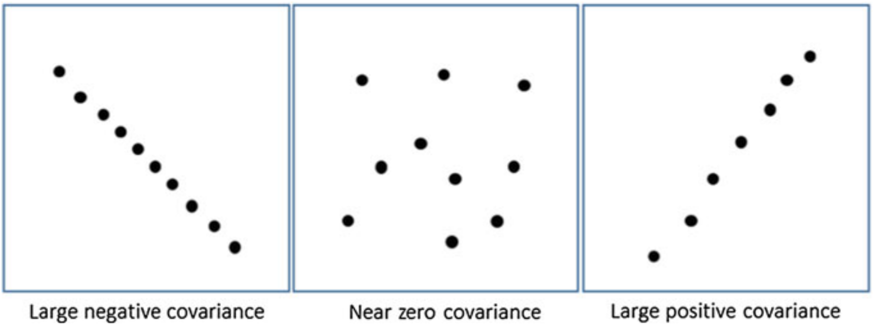


Fig. 15.2 Examples of covariance for three different data sets

covariance. It makes it impossible for example to compare covariances from data sets with different scales (e.g. pounds and inches). This issue can be fixed by dividing the covariance by the product of the standard deviation of each random variable, which gives Pearson's correlation coefficient.

Correlation is therefore a scaled version of covariance, used to assess the linear relationship between two variables and is calculated using the formula below.

$$Cor(x, y) = \frac{Cov(x, y)}{s_x s_y} \tag{15.4}$$

where $Cov(x, y)$ is the covariance between x and y and s_x, s_y are the sample standard deviations of x and y.

The significance of the correlation coefficient between two normally distributed variables can be evaluated using Fisher's z transformation (see the cor.test function in R for more details). Other tests exist for measuring the non-parametric relationship between two variables, such as Spearman's rho or Kendall's tau.

15.2.3 Graphical EDA

Univariate Graphical EDA

Histograms

Histograms are among the most useful EDA techniques, and allow you to gain insight into your data, including distribution, central tendency, spread, modality and outliers.

Histograms are bar plots of counts versus subgroups of an exposure variable. Each bar represents the frequency (count) or proportion (count divided by total count) of cases for a range of values. The range of data for each bar is called a bin. Histograms give an immediate impression of the shape of the distribution (symmetrical, uni/multimodal, skewed, outliers…). The number of bins heavily influences the final aspect of the histogram; a good practice is to try different values, generally from 10 to 50. Some examples of histograms are shown below as well as in the case studies. Please refer to the function called "Density" in the GitHub repository for this book (see code appendix at the end of this Chapter) (Figs. 15.3 and 15.4).

Histograms enable to confirm that an operation on data was successful. For example, if you need to log-transform a data set, it is interesting to plot the histogram of the distribution of the data before and after the operation (Fig. 15.5).

Histograms are interesting for finding outliers. For example, pulse oximetry can be expressed in fractions (range between 0 and 1) or percentage, in medical records. Figure 15.6 is an example of a histogram showing the distribution of pulse oximetry, clearly showing the presence of outliers expressed in a fraction rather than as a percentage.

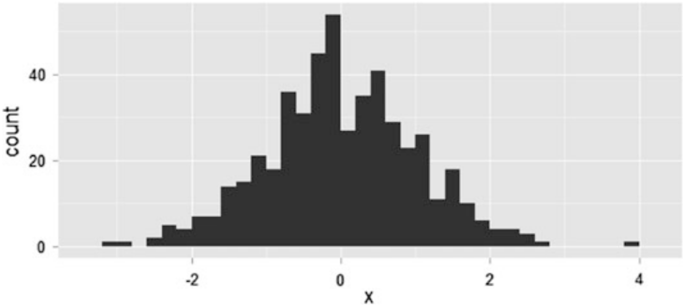


Fig. 15.3 Example of histogram

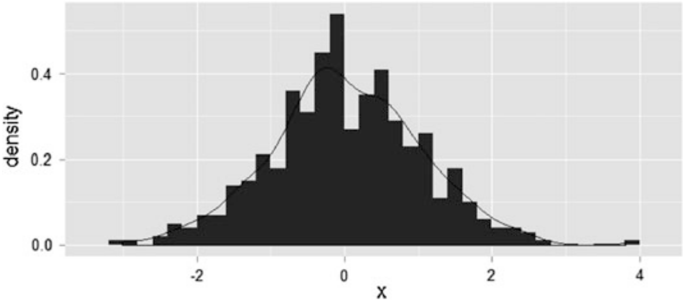


Fig. 15.4 Example of histogram with density estimate

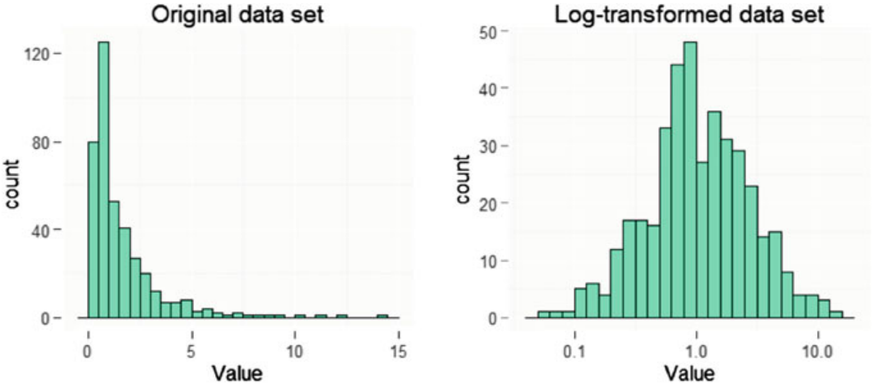


Fig. 15.5 Example of the effect of a log transformation on the distribution of the dataset

Stem Plots

Stem and leaf plots (also called stem plots) are a simple substitution for histograms. They show all data values and the shape of the distribution. For an example, Please refer to the function called “Stem Plot” in the GitHub repository for this book (see code appendix at the end of this Chapter) (Fig. 15.7).

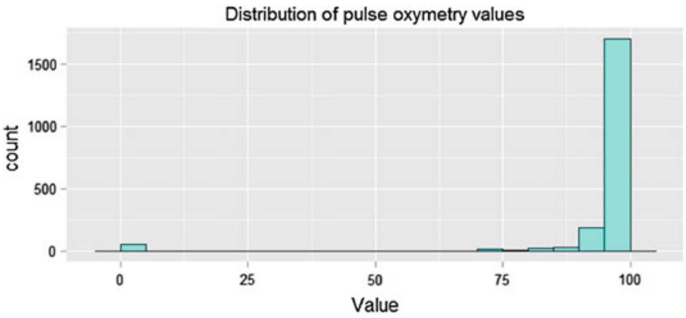


Fig. 15.6 Distribution of pulse oximetry

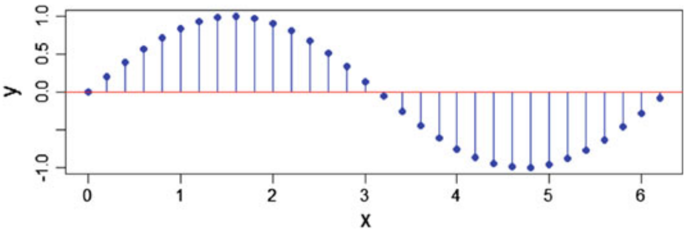


Fig. 15.7 Example of stem plot

Boxplots

Boxplots are interesting for representing information about the central tendency, symmetry, skew and outliers, but they can hide some aspects of the data such as multimodality. Boxplots are an excellent EDA technique because they rely on robust statistics like median and IQR.

Figure 15.8 shows an annotated boxplot which explains how it is constructed. The central rectangle is limited by Q1 and Q3, with the middle line representing the median of the data. The whiskers are drawn, in each direction, to the most extreme point that is less than 1.5 IQR beyond the corresponding hinge. Values beyond 1.5 IQR are considered outliers.

The "outliers" identified by a boxplot, which could be called "boxplot outliers" are defined as any points more than 1.5 IQRs above Q3 or more than 1.5 IQRs below Q1. This does not by itself indicate a problem with those data points. Boxplots are an exploratory technique, and you should consider designation as a boxplot outlier as just a suggestion that the points might be mistakes or otherwise unusual. Also, points not designated as boxplot outliers may also be mistakes. It is also important to realize that the number of boxplot outliers depends strongly on the size of the sample. In fact, for data that is perfectly normally distributed, we expect 0.70 % (about 1 in 140 cases) to be "boxplot outliers", with approximately half in either direction.

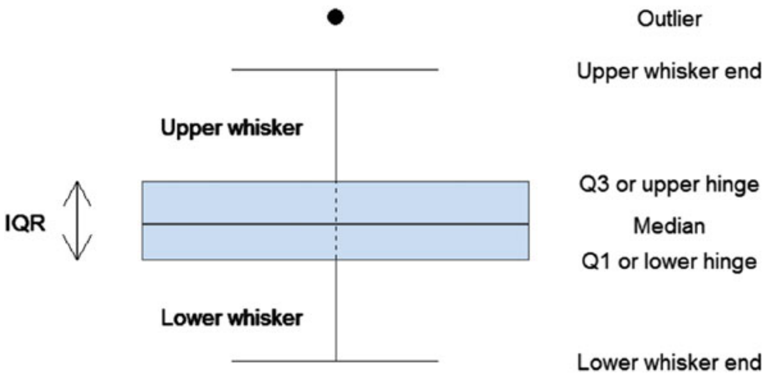


Fig. 15.8 Example of boxplot with annotations

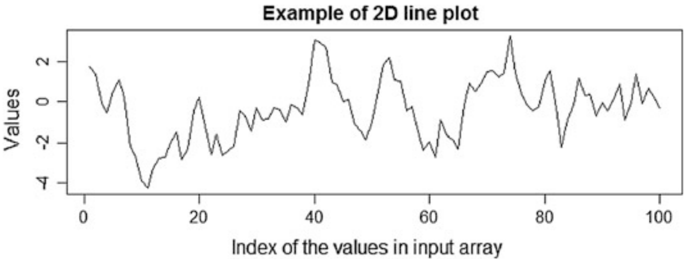


Fig. 15.9 Example of 2D line plot

2D Line Plot

2D line plots represent graphically the values of an array on the y-axis, at regular intervals on the x-axis (Fig. 15.9).

Probability Plots (Quantile-Normal Plot/QN Plot, Quantile-Quantile Plot/QQ Plot)

Probability plots are a graphical test for assessing if some data follows a particular distribution. They are most often used for testing the normality of a data set, as many statistical tests have the assumption that the exposure variables are approximately normally distributed. These plots are also used to examine residuals in models that rely on the assumption of normality of the residuals (ANOVA or regression analysis for example).

The interpretation of a QN plot is visual (Fig. 15.10): either the points fall randomly around the line (data set normally distributed) or they follow a curved pattern instead of following the line (non-normality). QN plots are also useful to identify skewness, kurtosis, fat tails, outliers, bimodality etc.

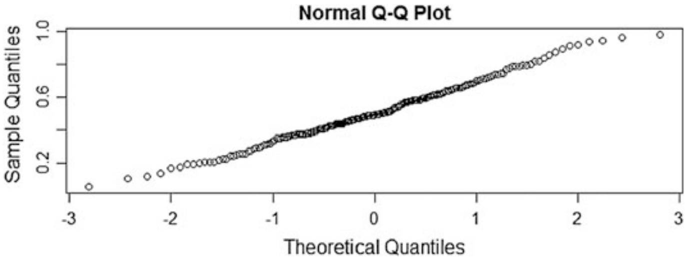


Fig. 15.10 Example of QQ plot

Besides the probability plots, there are many quantitative statistical tests (not graphical) for testing for normality, such as Pearson Chi^2, Shapiro-Wilk, and Kolmogorov-Smirnov.

Deviation of the observed distribution from normal makes many powerful statistical tools useless. Note that some data sets can be transformed to a more normal distribution, in particular with log-transformation and square-root transformations. If a data set is severely skewed, another option is to discretize its values into a finite set.

Multivariate Graphical EDA

Side-by-Side Boxplots

Representing several boxplots side by side allows easy comparison of the characteristics of several groups of data (example Fig. 15.11). An example of such boxplot is shown in the case study.

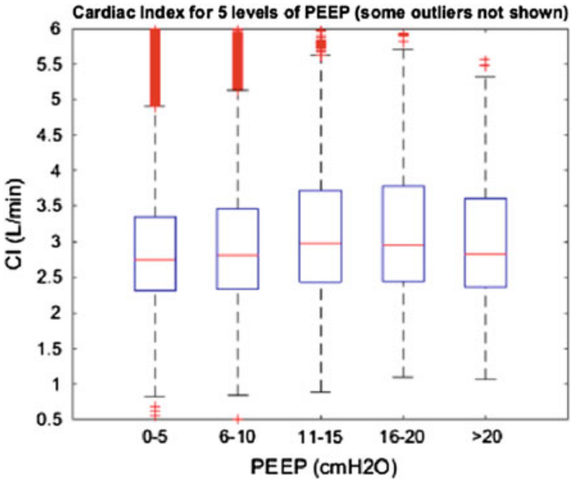


Fig. 15.11 Side-by-side boxplot showing the cardiac index for five levels of Positive end-expiratory pressure (PEEP)

Scatterplots

Scatterplots are built using two continuous, ordinal or discrete quantitative variables (Fig. 15.12). Each data point's coordinate corresponds to a variable. They can be complexified to up to five dimensions using other variables by differentiating the data points' size, shape or color.

Scatterplots can also be used to represent high-dimensional data in 2 or 3D (Fig. 15.13), using T-distributed stochastic neighbor embedding (t-SNE) or principal component analysis (PCA). t-SNE and PCA are dimension reduction features used to reduce complex data set in two (t-SNE) or more (PCA) dimensions.

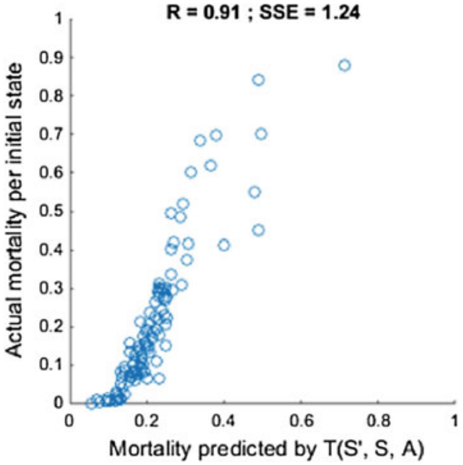


Fig. 15.12 Scatterpolot showing an example of actual mortality per rate of predicted mortality

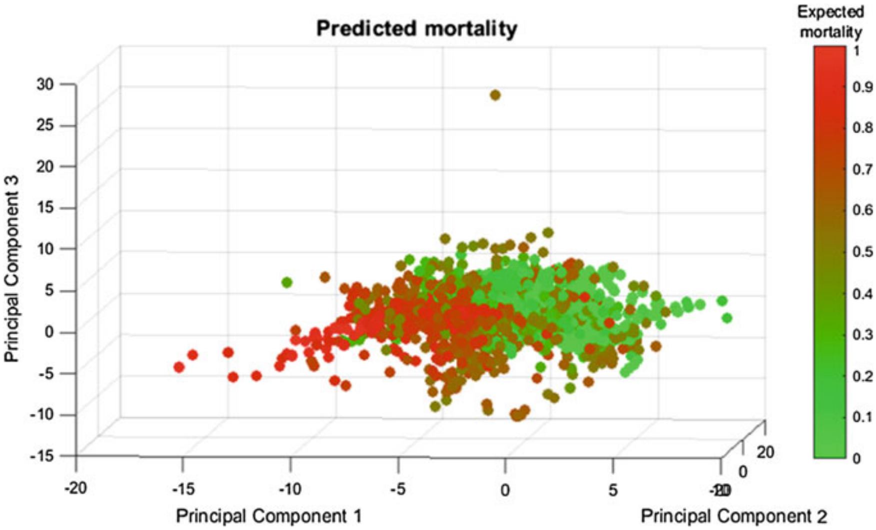


Fig. 15.13 3D representation of the first three dimension of a PCA

For binary variables (e.g. 28-day mortality vs. SOFA score), 2D scatterplots are not very helpful (Fig. 15.14, left). By dividing the data set in groups (in our example: one group per SOFA point), and plotting the average value of the outcome in each group, scatterplots become a very powerful tool, capable for example to identify a relationship between a variable and an outcome (Fig. 15.14, right).

Curve Fitting

Curve fitting is one way to quantify the relationship between two variables or the change in values over time (Fig. 15.15). The most common method for curve fitting relies on minimizing the sum of squared errors (SSE) between the data and the

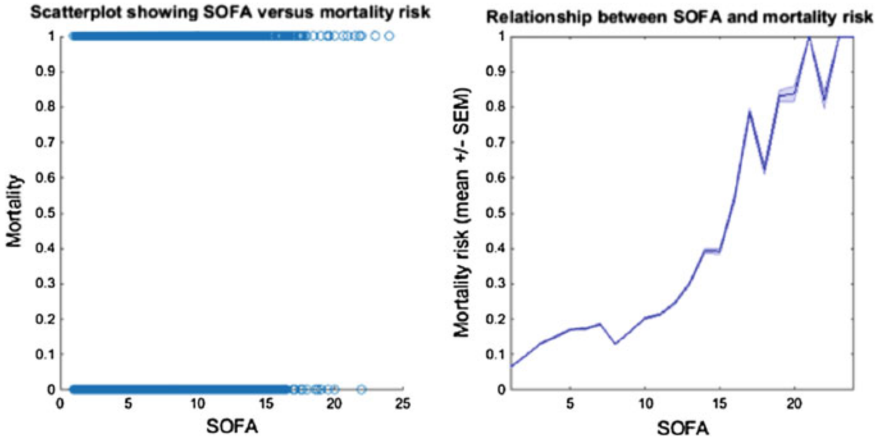


Fig. 15.14 Graphs of SOFA versus mortality risk

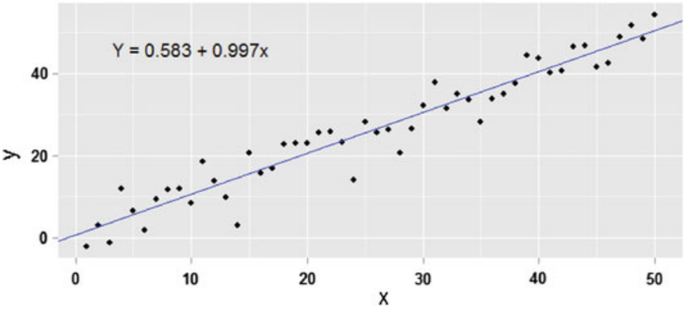


Fig. 15.15 Example of linear regression

fitted function. Please refer to the “Linear Fit” function to create linear regression slopes in R.

More Complicated Relationships

Many real life phenomena are not adequately explained by a straight-line relationship. An always increasing set of methods and algorithms exist to deal with that issue. Among the most common:

- Adding transformed explanatory variables, for example, adding x^2 or x^3 to the model.
- Using other algorithms to handle more complex relationships between variables (e.g., generalized additive models, spline regression, support vector machines, etc.).

Heat Maps and 3D Surface Plots

Heat maps are simply a 2D grid built from a 2D array, whose color depends on the value of each cell. The data set must correspond to a 2D array whose cells contain the values of the outcome variable. This technique is useful when you want to represent the change of an outcome variable (e.g. length of stay) as a function of two other variables (e.g. age and SOFA score).

The color mapping can be customized (e.g. rainbow or grayscale). Interestingly, the Matlab function *imagesc* scales the data to the full colormap range. Their 3D equivalent is mesh plots or surface plots (Fig. 15.16).

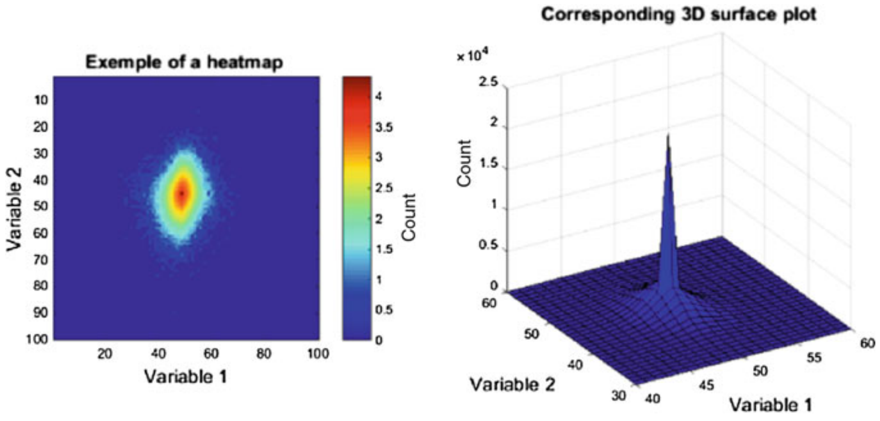


Fig. 15.16 Heat map (*left*) and surface plot (*right*)

15.3 Part 2—Case Study

This case study refers to the research that evaluated the effect of the placement of indwelling arterial catheters (IACs) in hemodynamically stable patients with respiratory failure in intensive care, from the MIMIC-II database.

For this case study, several aspects of EDA were used:

- The categorical data was first tabulated.
- Summary statistics were then generated to describe the variables of interest.
- Graphical EDA was used to generate histograms to visualize the data of interest.

15.3.1 Non-graphical EDA

Tabulation

To analyze, visualize and test for association or independence of categorical variables, they must first be tabulated. When generating tables, any missing data will be counted in a separate "NA" ("Not Available") category. Please refer to the Chap. 13 "Missing Data" for approaches in managing this problem. There are several methods for creating frequency or contingency tables in R, such as for example, tabulating outcome variables for mortality, as demonstrated in the case study. Refer to the "Tabulate" function found in the GitHub repository for this book (see code appendix at the end of this Chapter) for details on how to compute frequencies of outcomes for different variables.

Statistical Tests

Multiple statistical tests are available in R and we refer the reader to the Chap. 16 "Data Analysis" for additional information on use of relevant tests in R. For examples of a simple Chi-square…" as "For examples of a simple Chi-squared test, please refer to the "Chi-squared" function found in the GitHub repository for this book (see code appendix at the end of this Chapter). In our example, the hypothesis of independence between expiration in ICU and IAC is accepted ($p > 0.05$). On the contrary, the dependence link between day-28 mortality and IAC is rejected.

Summary statistics

Summary statistics as described above include, frequency, mean, median, mode, range, interquartile range, maximum and minimum values. An extract of summary statistics of patient demographics, vital signs, laboratory results and comorbidities, is shown in Table 6. Please refer to the function called "EDA Summary" in the

Table 15.5 Comparison between the two study cohorts (subsample of variables only)

Variables	Entire Cohort (N = 1776)		
	Non-IAC	IAC	*p*-value
Size	984 (55.4 %)	792 (44.6 %)	NA
Age (year)	51 (35–72)	56 (40–73)	0.009
Gender (female)	344 (43.5 %)	406 (41.3 %)	0.4
Weight (kg)	76 (65–90)	78 (67–90)	0.08
SOFA score	5 (4–6)	6 (5–8)	<0.0001
Co-morbidities			
CHF	97 (12.5 %)	116 (11.8 %)	0.7
…	…	…	…
Lab tests			
WBC	10.6 (7.8–14.3)	11.8 (8.5–15.9)	<0.0001
Hemoglobin (g/dL)	13 (11.3–14.4)	12.6 (11–14.1)	0.003
…	…	…	…

GitHub repository for this book (see code appendix at the end of this Chapter) (Table 15.5).

When separate cohorts are generated based on a common variable, in this case the presence of an indwelling arterial catheter, summary statistics are presented for each cohort.

It is important to identify any differences in subject baseline characteristics. The benefits of this are two-fold: first it is useful to identify potentially confounding variables that contribute to an outcome in addition to the predictor (exposure) variable. For example, if mortality is the outcome variable then differences in severity of illness between cohorts may wholly or partially account for any variance in mortality. Identifying these variables is important as it is possible to attempt to control for these using adjustment methods such as multivariable logistic regression. Secondly, it may allow the identification of variables that are associated with the predictor variable enriching our understanding of the phenomenon we are observing.

The analytical extension of identifying any differences using medians, means and data visualization is to test for statistically significant differences in any given subject characteristic using for example Wilcoxon-Rank sum test. Refer to Chap. 16 for further details in hypothesis testing.

15.3.2 Graphical EDA

Graphical representation of the dataset of interest is the principle feature of exploratory analysis.

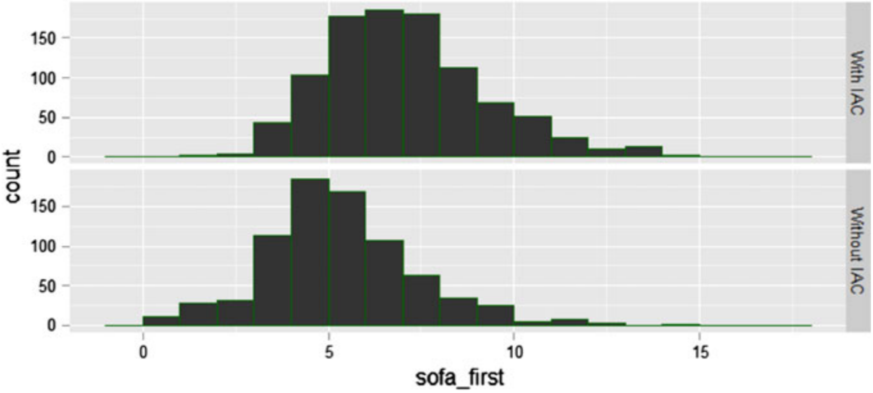


Fig. 15.17 histograms of SOFA scores by intra-arterial catheter status

Histograms

Histograms are considered the backbone of EDA for continuous data. They can be used to help the researcher understand continuous variables and provide key information such as their distribution. Outlined in *noise and outliers,* the histogram allows the researcher to visualize where the bulk of the data points are placed between the maximum and minimum values. Histograms can also allow a visual comparison of a variable between cohorts. For example, to compare severity of illness between patient cohorts, histograms of SOFA score can be plotted side by side (Fig. 15.17). An example of this is given in the code for this chapter using the "side-by-side histogram" function (see code appendix at the end of this Chapter).

Boxplot and ANOVA

Outside of the scope of this case study, the user may be interested in analysis of variance. When performing EDA and effective way to visualize this is through the use of boxplot. For example, to explore differences in blood pressure based on severity of illness subjects could be categorized by severity of illness with blood pressure values at baseline plotted (Fig. 15.18). Please refer to the function called "Box Plot" in the GitHub repository for this book (see code appendix at the end of this Chapter).

The box plot shows a few outliers which may be interesting to explore individually, and that people with a high SOFA score (>10) tend to have a lower blood pressure than people with a lower SOFA score.

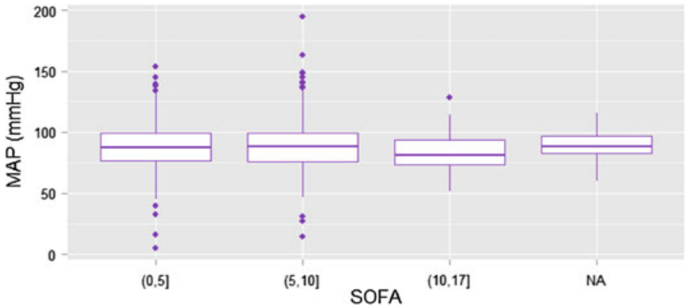


Fig. 15.18 Side-by-side boxplot of MAP for different levels of severity at admission

15.4 Conclusion

In summary, EDA is an essential step in many types of research but is of particular use when analyzing electronic health care records. The tools described in this chapter should allow the researcher to better understand the features of a dataset and also to generate novel hypotheses.

Take Home Messages

1. Always start by exploring a dataset with an open mind for discovery.
2. EDA allows to better apprehend the features and possible issues of a dataset.
3. EDA is a key step in generating research hypothesis.

Code Appendix

The code used in this chapter is available in the GitHub repository for this book: https://github.com/MIT-LCP/critical-data-book. Further information on the code is available from this website.

References

1. Natrella M (2010) NIST/SEMATECH e-Handbook of Statistical Methods. NIST/SEMATECH
2. Mosteller F, Tukey JW (1977) Data analysis and regression. Addison-Wesley Pub. Co., Boston
3. Tukey J (1977) Exploratory data analysis. Pearson, London
4. Seltman HJ (2012) Experimental design and analysis. Online http://www.stat.cmu.edu/∼hseltman/309/Book/Book.pdf
5. Kaski, Samuel (1997) "Data exploration using self-organizing maps." *Acta polytechnica scandinavica: Mathematics, computing and management in engineering series no. 82. 1997*.
6. Hill T, Lewicki P (2006) Statistics: methods and applications: a comprehensive reference for science, industry, and data mining. StatSoft, Inc., Tulsa
7. CRAN (2016) The Comprehensive R archive network—packages. Contributed Packages, 10 Jan 2016 [Online]. Available: https://cran.r-project.org/web/packages/. Accessed: 10 Jan 2016
8. Grubbs F (1969) Procedures for detecting outlying observations in samples. Technometrics 11(1)
9. Joanes DN, Gill CA (1998) Comparing measures of sample skewness and kurtosis. The Statistician 47:183–189.

Chapter 16
Data Analysis

Jesse D. Raffa, Marzyeh Ghassemi, Tristan Naumann, Mengling Feng and Douglas Hsu

Learning Objectives

- Understand how the study objective and data types determine the type of data analysis.
- Understand the basics of the three most common analysis techniques used in the studies involving health data.
- Execute a case study to fulfil the study objective, and interpret the results.

16.1 Introduction to Data Analysis

16.1.1 Introduction

This chapter presents an overview of data analysis for health data. We give a brief introduction to some of the most common methods for data analysis of health care data, focusing on choosing appropriate methodology for different types of study objectives, and on presentation and the interpretation of data analysis generated from health data. We will provide an overview of three very powerful analysis methods: linear regression, logistic regression and Cox proportional hazards models, which provide the foundation for most data analysis conducted in clinical studies.

Chapter Goals
By the time you complete this chapter you should be able to:

1. Understand how different study objectives will influence the type of data analysis (Sect. 16.1)
2. Be able to carry out three different types of data analysis that are common for health data (Sects. 16.2–16.4).
3. Present and interpret the results of these analyses types (Sects. 16.2–16.4)

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_16

4. Understand the limitations and assumptions underlying the different types of analyses (Sects. 16.2–16.4).
5. Replicate an analysis from a case study using some of the methods learned in the chapter (Sect. 16.5)

Outline

This chapter is composed of five sections. First, in this section we will cover identifying data types and study objectives. These topics will enable us to pick an appropriate analysis method among linear (Sect. 16.2) or logistic (Sect. 16.3) regression, and survival analysis (Sect. 16.4), which comprise the next three sections. Following that, we will use what we learned on a case study using real data from Medical Information Mart for Intensive Care II (MIMIC-II), briefly discuss model building and finally, summarize what we have learned (Sect. 16.5)

16.1.2 Identifying Data Types and Study Objectives

In this section we will examine how different study objectives and data types affect the approaches one takes for data analysis. Understanding the data structure and study objective is likely the most important aspect to choosing an appropriate analysis technique.

Study Objectives

Identifying the study objective is an extremely important aspect of planning data analysis for health data. A vague or poorly described objective often leads to a poorly executed analysis. The study objective should clearly identify the study population, the outcome of interest, the covariate(s) of interest, the relevant time points of the study, and what you would like to do with these items. Investing time to make the objective very specific and clear often will save time in the long run.

An example of a clearly stated study objective would be:

> To estimate the reduction in 28 day mortality associated with vasopressor use during the first three days from admission to the MICU in MIMIC II.

An example of a vague and difficult to execute study objective may be:

> To predict mortality in ICU patients.

While both may be trying to accomplish the same goal, the first gives a much clearer path for the data scientist to perform the necessary analysis, as it identifies the study population (those admitted to the MICU in MIMIC II), outcome (28 day mortality), covariate of interest (vasopressor use in the first three days of the MICU admission), relevant time points (28 days for the outcome, within the first three days for the covariate). The objective does not need to be overly complicated, and it's often convenient to specify primary and secondary objectives, rather than an overly complex single objective.

Data Types

After specifying a clear study objective, the next step is to determine the types of data one is dealing with. The first distinction is between outcomes and covariates. Outcomes are what the study aims to investigate, improve or affect. In the above example of a clearly stated objective, our outcome is 28 day mortality. Outcomes are also sometimes referred to as response or dependent variables. Covariates are the variables you would like to study for their effect on the outcome, or believe may have some nuisance effect on the outcome you would like to control for. Covariates also go by several different names, including: features, predictors, independent variables and explanatory variables. In our example objective, the primary covariate of interest is vasopressor use, but other covariates may also be important in affecting 28 day mortality, including age, gender, and so on.

Once you have identified the study outcomes and covariates, determining the data types of the outcomes will often be critical in choosing an appropriate analysis technique. Data types can generally be identified as either continuous or discrete. Continuous variables are those which can plausibly take on any numeric (real number) value, although this requirement is often not explicitly met. This contrasts with discrete data, which usually takes on only a few values. For instance, gender can take on two values: male or female. This is a *binary* variable as it takes on two values. More discussion on data types can be found in Chap. 11.

There is a special type of data which can be considered simultaneously as continuous and discrete types, as it has two components. This frequently occurs in time to event data for outcomes like mortality, where both the occurrence of death and the length of survival are of interest. In this case, the discrete component is if the event (e.g., death) occurred during the observation period, and the continuous component is the time at which death occurred. The time at which the death occurred is not always available: in this case the time of the last observation is used, and the data is partially *censored*. We discuss censoring in more detail later in Sect. 16.4.

Figure 16.1 outlines the typical process by which you can identify outcomes from covariates, and determine which type of data type your outcome is. For each of the types of outcomes we highlighted—continuous, binary and survival, there are a set of analysis methods that are most common for use in health data—linear regression, logistic regression and Cox proportional hazards models, respectively.

Other Important Considerations

The discussion thus far has given a basic outline of how to choose an analysis method for a given study objective. Some caution is merited as this discussion has been rather brief and while it covers some of the most frequently used methods for analyzing health data, it is certainly not exhaustive. There are many situations where this framework and subsequent discussion will break down and other methods will be necessary. In particular, we highlight the following situations:

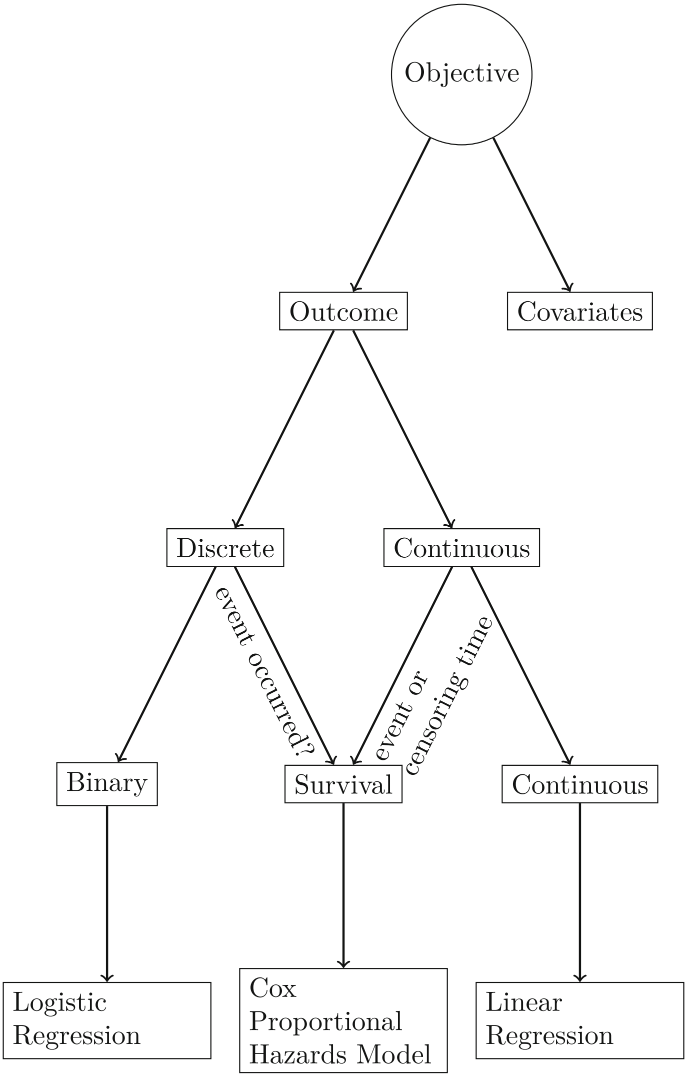


Fig. 16.1 Flow diagram of simplified process for choosing an analysis method based on the study objective and outcome data types

1. When the data is not patient level data, such as aggregated data (totals) instead of individual level data.
2. When patients contribute more than one observation (i.e., outcome) to the dataset.

In these cases, other techniques should be used.

16.1.3 Case Study Data

We will be using a case study [1] to explore data analysis approaches in health data. The case study data originates from a study examining the effect of indwelling arterial catheters (IAC) on 28 day mortality in the intensive care unit (ICU) in patients who were mechanically ventilated during the first day of ICU admission. The data comes from MIMIC II v2.6. At this point you are ready to do data analysis (the data extraction and cleaning has already been completed) and we will be using a comma separated (.csv) file generated after this process, which you can load directly off of PhysioNet [2, 3]:

```
url <- "http://physionet.org/physiobank/database/mimic2-iaccd/full_cohort_data.csv";
dat <- read.csv(url)
# Or download the csv file from:
# http://physionet.org/physiobank/database/mimic2-iaccd/full_cohort_data.csv
# Type: dat <- read.csv(file.choose())
# And navigate to the file you downloaded (likely in your download directory)
```

The header of this file with the variable names can be accessed using the `names` function in `R`.

```
names(dat)
```

```
##  [1] "aline_flg"          "icu_los_day"        "hospital_los_day"
##  [4] "age"                "gender_num"         "weight_first"
##  [7] "bmi"                "sapsi_first"        "sofa_first"
## [10] "service_unit"       "service_num"        "day_icu_intime"
## [13] "day_icu_intime_num" "hour_icu_intime"    "hosp_exp_flg"
## [16] "icu_exp_flg"        "day_28_flg"         "mort_day_censored"
## [19] "censor_flg"         "sepsis_flg"         "chf_flg"
## [22] "afib_flg"           "renal_flg"          "liver_flg"
## [25] "copd_flg"           "cad_flg"            "stroke_flg"
## [28] "mal_flg"            "resp_flg"           "map_1st"
## [31] "hr_1st"             "temp_1st"           "spo2_1st"
## [34] "abg_count"          "wbc_first"          "hgb_first"
## [37] "platelet_first"     "sodium_first"       "potassium_first"
## [40] "tco2_first"         "chloride_first"     "bun_first"
## [43] "creatinine_first"   "po2_first"          "pco2_first"
## [46] "iv_day_1"
```

There are 46 variables listed. The primary focus of the study was on the effect that IAC placement (`aline_flg`) has on 28 day mortality (`day_28_flg`). After we have covered the basics, we will identify a research objective and an appropriate analysis technique, and execute an abbreviated analysis to illustrate how to use these techniques to address real scientific questions. Before we do this, we need to cover the basic techniques, and we will introduce three powerful data analysis methods frequently used in the analysis of health data. We will use examples from the case study dataset to introduce these concepts, and will return to the the question of the effect of IAC has on mortality towards the end of thischapter.

16.2 Linear Regression

16.2.1 Section Goals

In this section, the reader will learn the fundamentals of linear regression, and how to present and interpret such an analysis.

16.2.2 Introduction

Linear regression provides the foundation for many types of analyses we perform on health data. In the simplest scenario, we try to relate one continuous outcome, y, to a single continuous covariate, x, by trying to find values for β_0 and β_1 so that the following equation:

$$y = \beta_0 + \beta_1 \times x$$

fits the data 'optimally'.[1] We call these optimal values: $\hat{\beta}_0$ and $\hat{\beta}_1$ to distinguish them from the true values of β_0 and β_1 which are often unknowable. In Fig. 16.2, we see a scatter plot of TCO2 (y: outcome) levels versus PCO2 (x: covariate) levels. We can clearly see that as PCO2 levels increase, the TCO2 levels also increase. This would suggest that we may be able to fit a linear regression model which predicts TCO2 from PCO2.

It is always a good idea to visualize the data when you can, which allows one to assess if the subsequent analysis corresponds to what you could see with your eyes. In this case, a scatter plot can be produced using the `plot` function:

```r
plot(dat$pco2_first,dat$tco2_first,xlab="PCO2",ylab="TCO2",pch=19,xlim=c(0,175))
```

which produces the scattered points in Fig. 16.2.

Finding the best fit line for the scatter plot in Fig. 16.2 in `R` is relatively straightforward:

[1] Exactly what optimally means is beyond the scope of this chapter, but for those who are interested, we are trying to find values of β_0 and β_1 which minimize the squared distance between the fitted line and the observed data point, summed over all data points. This quantity is known as sum of squares error, or when divided by the number of observations is known as the mean squared error.

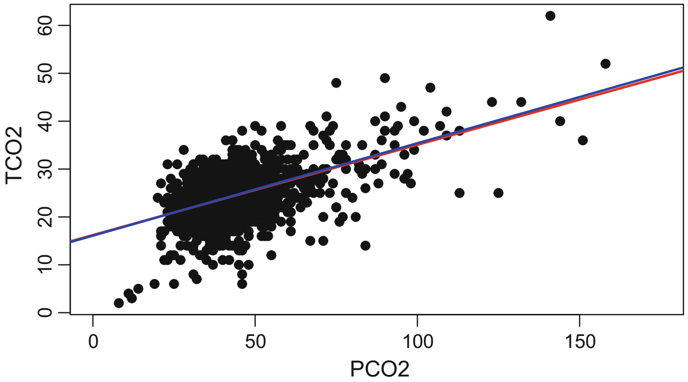


Fig. 16.2 Scatterplot of PCO2 (x-axis) and TCO2 (y-axis) along with linear regression estimates from the quadratic model (co2.quad.lm) and linear only model (co2.lm)

```
co2.lm <- lm(tco2_first ~ pco2_first,data=dat)
```

Dissecting this command from left to right. The co2.lm <- part assigns the right part of the command to a new variable or object called co2.lm which contains information relevant to our linear regression model. The right side of this command runs the lm function in R. lm is a powerful function in R that fits linear models. As with any command in R, you can find additional help information by running ?lm from the R command prompt. The basic lm command has two parts. The first is the formula which has the general syntax outcome ~ covariates. Here, our outcome variable is called tco2_first and we are just fitting one covariate, pco2_first, so our formula is tco2_first ~ pco2_first. The second argument is separated by a comma and is specifying the data frame to use. In our case, the data frame is called dat, so we pass data = dat, noting that both tco2_first and pco2_first are columns in the dataframe dat. The overall procedure of specifying a model formula (tco2_first ~ pco2_first), a data frame (data = dat) and passing it an appropriate R function (lm) will be used throughout this chapter, and is the foundation for many types of statistical modeling in R.

We would like to see some information about the model we just fit, and often a good way of doing this is to run the summary command on the object we created:

```
summary(co2.lm)
```

```
##
## Call:
## lm(formula = tco2_first ~ pco2_first, data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.8852  -2.5080   0.1891   2.8077  19.2005
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.210859   0.359676   45.07   <2e-16 ***
## pco2_first   0.188572   0.007886   23.91   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.395 on 1588 degrees of freedom
##   (186 observations deleted due to missingness)
## Multiple R-squared:  0.2647, Adjusted R-squared:  0.2643
## F-statistic: 571.8 on 1 and 1588 DF,  p-value: < 2.2e-16
```

This outputs information about the `lm` object we created in the previous step. The first part recalls the model we fit, which is useful when we have fit many models, and are trying to compare them. The second part lists some summary information about what are called residuals—an important topic for validating modeling assumptions covered in [8]. Next lists the coefficient estimates—these are the $\hat{\beta}_0$, `(Intercept)`, and $\hat{\beta}_1$, `pco2_first`, parameters in the best fit line we are trying to estimate. This output is telling us that the best fit equation for the data is:

$$\texttt{tco2\_first} = 16.21 + 0.189 \times \texttt{pco2\_first}.$$

These two quantities have important interpretations. The estimated intercept ($\hat{\beta}_0$) tells us what TCO2 level we would predict for an individual with a PCO2 level of 0. This is the mathematical interpretation, and often this quantity has limited practical use. The estimated slope ($\hat{\beta}_1$) on the other hand can be interpreted as how quickly the predicted value of TCO2 goes up for every unit increase in PCO2. In this case, we estimate that TCO2 goes up about 0.189 mmol/L for every 1 mm Hg increase in PCO2. Each coefficient estimate has a corresponding `Std. Error` (standard error). This is a measure of how certain we are about the estimate. If the standard error is large relative to the coefficient then we are less certain about our estimate. Many things can affect the standard error, including the study sample size. The next column in this table is the `t value`, which is simply the coefficient estimate divided by the standard error. This is followed by `Pr(>|t|)` which is also known as the *p*-value. The last two quantities are relevant to an area of statistics called hypothesis testing which we will cover briefly now.

Hypothesis Testing

Hypothesis testing in statistics is fundamentally about evaluating two competing hypotheses. One hypothesis, called the *null hypothesis* is setup as a straw man (a sham argument set up to be defeated), and is the hypothesis you would like to provide evidence *against*. In the analysis methods we will discuss in this chapter, this is almost always $\beta_k = 0$, and it is often written as $H_0 : \beta_k = 0$. The alternative (second) hypothesis is commonly assumed to be $\beta_k \neq 0$, and will often be written as $H_A : \beta_k \neq 0$. A statistical significance level, α, should be established before any analysis is performed. This value is known as the Type I error, and is the probability of rejecting the null hypothesis when the null hypothesis is true, i.e. of incorrectly concluding that the null hypothesis is false. In our case, it is the probability that we falsely conclude that the coefficient is non-zero, when the coefficient is actually zero. It is common to set the Type I error at 0.05.

After specifying the null and alternative hypotheses, along with the significance level, hypotheses can be tested by computing a p-value. The actual computation of p-values is beyond the scope of this chapter, but we will cover the interpretation and provide some intuition. P-values are the probability of observing data as extreme or more extreme than what was seen, assuming the null hypothesis is *true*. The null hypothesis is $\beta_k = 0$, so when would this be unlikely? It is probably unlikely when we estimate β_k to be rather large. However, how large is large enough? This would likely depend on how certain we are about the estimate of β_k. If we were very certain, $\hat{\beta}_k$ likely would not have to be very large, but if we are less certain, then we might not think it to be unlikely for even very large values of $\hat{\beta}_k$. A p-value balances both of these aspects, and computes a single number. We reject the null hypothesis when the p-value is smaller than the significance level, α.

Returning to our fit model, we see that the p-value for both coefficients are tiny (`<2e-16`), and we would reject both null hypotheses, concluding that neither coefficient is likely zero. What do these two hypotheses mean at a practical level? The intercept being zero, $\beta_0 = 0$ would imply the best fit line goes through the origin [the (x, y) point (0, 0)], and we would reject this hypothesis. The slope being zero would mean that the best fit line would be a flat horizontal line, and did not increase as PCO2 increases. Clearly there is a relationship between TCO2 and PCO2, so we would also reject this hypothesis. In summary, we would conclude that we need both an intercept and a slope in the model. A next obvious question would be, could the relationship be more complicated than a straight line? We will examine this next.

16.2.3 Model Selection

Model selection are techniques related to selecting the best model from a list (perhaps rather large list) of candidate models. We will cover some basics here, as more complicated techniques will be covered in a later chapter. In the simplest case, we have two models, and we want to know which one we should use.

We will begin by examining if the relationship between TCO2 and PCO2 is more complicated than the model we fit in the previous section. If you recall, we fit a model where we considered a linear pco2_first term: tco2_first = $\beta_0 + \beta_1 \times$ pco2_first. One may wonder if including a quadratic term would fit the data better, i.e. whether:

$$\texttt{tco2\_first} = \beta_0 + \beta_1 \times \texttt{pco2\_first} + \beta_2 \times \texttt{pco2\_first}^2,$$

is a better model. One way to evaluate this is by testing the null hypothesis: $\beta_2 = 0$. We do this by fitting the above model, and looking at the output. Adding a quadratic term (or any other function) is quite easy using the lm function. It is best practice to enclose any of these functions in the I() function to make sure they get evaluated as you intended. The I() forces the formula to evaluate what is passed to it as is, as the ^ operator has a different use in formulas in R (see ?formula for further details). Fitting this model, and running the summary function for the model:

```r
co2.quad.lm <- lm(tco2_first ~ pco2_first + I(pco2_first^2),data=dat)
summary(co2.quad.lm)$coef
```

```
##                        Estimate   Std. Error    t value     Pr(>|t|)
## (Intercept)     16.0916260327 0.7713394026 20.8619266 1.309513e-85
## pco2_first       0.1930281243 0.0266927962  7.2314689 7.401248e-13
## I(pco2_first^2) -0.0000356873 0.0002042135 -0.1747548 8.612946e-01
```

You will note that we have abbreviated the output from the summary function by appending $coef to the summary function: this tells R we would like information about the coefficients only. Looking first at the estimates, we see the best fit line is estimated as:

$$\texttt{tco2\_first} = 160.09 + 0.19 \times \texttt{pco2\_first} + 0.00004 \times \texttt{pco2\_first}^2.$$

We can add both best fit lines to Fig. 16.2 using the abline function:

```r
abline(co2.lm,col='red')
abline(co2.quad.lm,col='blue')
```

and one can see that the red (linear term only) and blue (linear and quadratic terms) fits are nearly identical. This corresponds with the relatively small coefficient estimate for the I(pco2_first^2) term. The p-value for this coefficient is about 0.86, and at the 0.05 significance level we would likely conclude that a quadratic term is not necessary in our model to fit the data, as the linear term only model fits the data nearly as well.

Statistical Interactions and Testing Nested Models

We have concluded that a linear (straight line) model fit the data quite well, but thus far we have restricted our exploration to just one variable at a time. When we include other variables, we may wonder if the same straight line is true for all patients. For example, could the relationship between PCO2 and TCO2 be different among men and women? We could subset the data into a data frame for men and a data frame for women, and then fit separate regressions for each gender. Another more efficient way to accomplish this is by fitting both genders in a single model, and including gender as a covariate. For example, we may fit:

$$\texttt{tco2\_first} = \beta_0 + \beta_1 \times \texttt{pco2\_first} + \beta_2 \times \texttt{gender\_num}.$$

The variable `gender_num` takes on values 0 for women and 1 for men, and for men the model is:

$$\texttt{tco2\_first} = \underbrace{(\beta_0 + \beta_2)}_{\text{intercept}} + \beta_1 \times \texttt{pco2\_first},$$

and in women:

$$\texttt{tco2\_first} = \beta_0 + \beta_1 \times \texttt{pco2\_first}.$$

As one can see these models have the same slope, but different intercepts (the distance between the slopes is β_2). In other words, the lines fit for men and women will be parallel and be separated by a distance of β_2 for all values of `pco2_first`. This isn't exactly what we would like, as the slopes may also be different. To allow for this, we need to discuss the idea of an interaction between two variables. An interaction is essentially the product of two covariates. In this case, which we will call the interaction model, we would be fitting:

$$\texttt{tco2\_first} = \beta_0 + \beta_1 \times \texttt{pco2\_first} + \beta_2 \times \texttt{gender\_num} + \beta_3 \times \underbrace{\texttt{gender\_num} \times \texttt{pco2\_first}}_{\text{interaction term}}.$$

Again, separating the cases for men:

$$\texttt{tco2\_first} = \underbrace{(\beta_0 + \beta_2)}_{\text{intercept}} + \underbrace{(\beta_1 + \beta_3)}_{\text{slope}} \times \texttt{pco2\_first},$$

and women:

$$\texttt{tco2\_first} = \underbrace{(\beta_0)}_{\text{intercept}} + \underbrace{(\beta_1)}_{\text{slope}} \times \texttt{pco2\_first}.$$

Now men and women have different intercepts *and* slopes.

Fitting these models in R is relatively straightforward. Although not absolutely required in this particular circumstance, it is wise to make sure that R handles data types in the correct way by ensuring our variables are of the right class. In this particular case, men are coded as 1 and women as 0 (a discrete binary covariate) but R thinks this is numeric (continuous) data:

```
class(dat$gender_num)
```

```
## [1] "integer"
```

Leaving this unaltered, will not affect the analysis in this instance, but it can be problematic when dealing with other types of data such as categorical data with several categories (e.g., ethnicity). Also, by setting the data to the right type, the output R generates can also be more informative. We can set the gender_num variable to the class factor by using the as.factor function.

```
dat$gender_num <- as.factor(dat$gender_num)
```

Here we have just overwritten the old variable in the dat data frame with a new copy which is of class

`factor:`

```
class(dat$gender_num)
```

```
## [1] "factor"
```

Now that we have the gender variable correctly encoded, we can fit the models we discussed above. First the model with gender as a covariate, but no interaction. We can do this by simply adding the variable gender_num to the previous formula for our co2.lm model fit.

```
co2.gender.lm <- lm(tco2_first ~ pco2_first + gender_num,data=dat)
summary(co2.gender.lm)$coef
```

```
##               Estimate  Std. Error    t value      Pr(>|t|)
## (Intercept) 16.3043942 0.377712532 43.1661457 6.337240e-270
## pco2_first   0.1888542 0.007894741 23.9215128 3.015777e-108
## gender_num1 -0.1816540 0.223738366 -0.8119036  4.169687e-01
```

This output is very similar to what we had before, but now there's a gender_num term as well. The 1 is present in the first column after gender_num, and it tells us who this coefficient is relevant to (subjects with 1 for the gender_num – men). This is always relative to the baseline group, and in this case this is women.

The estimate is negative, meaning that the line fit for males will be below the line for females. Plotting this fit curve in Fig. 16.3:

```
plot(dat$pco2_first, dat$tco2_first, col = dat$gender_num, xlab = "PCO2", ylab = "TCO2",
    xlim = c(0, 40), type = "n", ylim = c(15, 25))
abline(a = c(coef(co2.gender.lm)[1]), b = coef(co2.gender.lm)[2])
abline(a = coef(co2.gender.lm)[1] + coef(co2.gender.lm)[3], b = coef(co2.gender.lm)[2],
    col = "red")
```

we see that the lines are parallel, but almost indistinguishable. In fact, this plot has been cropped in order to see any difference at all. From the estimate from the summary output above, the difference between the two lines is −0.182 mmol/L, which is quite small, so perhaps this isn't too surprising. We can also see in the above summary output that the *p*-value is about 0.42, and we would likely *not* reject the null hypothesis that the true value of the gender_num coefficient is zero.

And now moving on to the model with an interaction between pco2_first and gender_num. To add an interaction between two variables use the * operator within a model formula. By default, R will add all of the main effects (variables contained in the interaction) to the model as well, so simply adding pco2_first*gender_num will add effects for pco2_first and gender_num in addition to the interaction between them to the model fit.

```
co2.gender.interaction.lm <- lm(tco2_first ~ pco2_first*gender_num,data=dat)
summary(co2.gender.interaction.lm)$coef
```

```
##                          Estimate Std. Error   t value      Pr(>|t|)
## (Intercept)           15.85443226 0.48869107 32.442648 1.591490e-177
## pco2_first             0.19939518 0.01072876 18.585105  6.559901e-70
## gender_num1            0.81437833 0.72225677  1.127547  2.596819e-01
## pco2_first:gender_num1 -0.02297002 0.01583758 -1.450348  1.471591e-01
```

The estimated coefficients are $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2$ and $\hat{\beta}_3$, respectively, and we can determine the best fit lines for men:

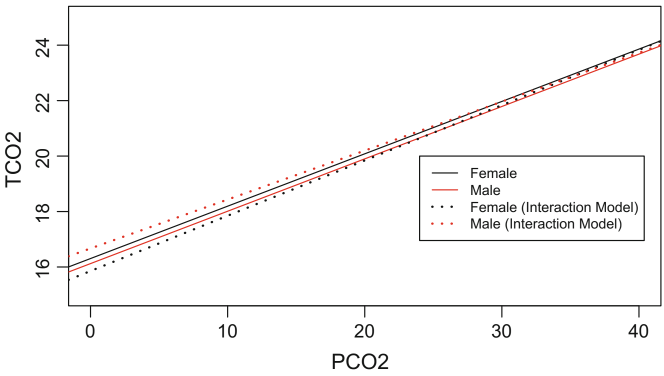


Fig. 16.3 Regression fits of PCO2 on TCO2 with gender (*black* female; *red* male; *solid* no interaction; *dotted* with interaction). *Note* Both axes are cropped for illustration purposes

$$\begin{aligned}\texttt{tco2\_first} &= (15.85 + 0.81) + (0.20 - 0.023) \times \texttt{pco2\_first}\\ &= 16.67 + 0.18 \times \texttt{pco2\_first},\end{aligned}$$

and for women:

$$\texttt{tco2\_first} = 15.85 + 0.20 \times \texttt{pco2\_first}.$$

Based on this, the men's intercept should be higher, but their slope should be not as steep, relative to the women. Let's check this and add the new model fits as dotted lines and add a legend to Fig. 16.3.

```
abline(a = coef(co2.gender.interaction.lm)[1], b = coef(co2.gender.interaction.lm)[2],
    lty = 3, lwd = 2)
abline(a = coef(co2.gender.interaction.lm)[1] + coef(co2.gender.interaction.lm)[3],
    b = coef(co2.gender.interaction.lm)[2] + coef(co2.gender.interaction.lm)[4],
    col = "red", lty = 3, lwd = 2)
legend(24, 20, lty = c(1, 1, 3, 3), lwd = c(1, 1, 2, 2), col = c("black", "red",
    "black", "red"), c("Female", "Male", "Female (Interaction Model)", "Male (Interaction Model)"))
```

We can see that the fits generated from this plot are a little different than the one generated for a model without the interaction. The biggest difference is that the dotted lines are no longer parallel. This has some serious implications, particularly when it comes to interpreting our result. First note that the estimated coefficient for the `gender_num` variable is now positive. This means that at `pco2_first = 0`, men (red) have higher `tco2_first` levels than women (black). If you recall in the previous model fit, women had higher levels of `tco2_first` at all levels of `pco2_first`. At some point around `pco2_first = 35` this changes and women (black) have higher `tco2_first` levels than men (red). This means that the effect of `gender_num` *may* vary as you change the level of `pco2_first`, and is why interactions are often referred to as effect modification in the epidemiological literature. The effect need not change signs (i.e., the lines do not need to cross) over the observed range of values for an interaction to be present.

The question remains, is the variable gender_num important? We looked at this briefly when we examined the t value column in the no interaction model which included gender_num. What if we wanted to test (simultaneously) the null hypothesis: β_2 *and* $\beta_3 = 0$. There is a useful test known as the F-test which can help us in this exact scenario where we want to look at if we should use a larger model (more covariates) or use a smaller model (fewer covariates). The F-test applies only to *nested models*—the larger model *must* contain each covariate that is used in the smaller model, and the smaller model *cannot* contain covariates which are not in the larger model. The interaction model and the model with gender are nested models since all the covariates in the model with gender are also in the larger interaction model. An example of a non-nested model would be the quadratic model and the interaction model: the smaller (quadratic) model has a term (pco2_first^2) which is not in the larger (interaction) model. An F-test would not be appropriate for this latter case.

To perform an F-test, first fit the two models you wish to consider, and then run the anova command passing the two model objects.

```
anova(co2.lm,co2.gender.interaction.lm)
```

```
## Analysis of Variance Table
##
## Model 1: tco2_first ~ pco2_first
## Model 2: tco2_first ~ pco2_first * gender_num
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1   1588 30674
## 2   1586 30621  2    53.349 1.3816 0.2515
```

As you can see, the anova command first lists the models it is considering. Much of the rest of the information is beyond the scope of this chapter, but we will highlight the reported F-test p-value (Pr(>F)), which in this case is 0.2515. In nested models, the null hypothesis is that all coefficients in the larger model and not in the smaller model are zero. In the case we are testing, our null hypothesis is β_2 *and* $\beta_3 = 0$. Since the p-value exceeds the typically used significance level ($\alpha = 0.05$), we would not reject the null hypothesis, and likely say the smaller model explains the data just as well as the larger model. If these were the only models we were considering, we would use the smaller model as our final model and report the final model in our results. We will now discuss what exactly you should report and how you can interpret the results.

16.2.4 Reporting and Interpreting Linear Regression

We will briefly discuss how to communicate a linear regression analysis. In general, before you present the results, some discussion of how you got the results should be done. It is a good idea to report: whether you transformed the outcome or any covariates in anyway (e.g., by taking the logarithm), what covariates you considered and how you chose the covariates which were in the model you reported. In our above example, we did not transform the outcome (TCO2), we considered PCO2 both as a linear and quadratic term, and we considered gender on its own and as an interaction term with PCO2. We first evaluated whether a quadratic term should be included in the model by using a t-test, after which we considered a model with gender and a gender-PCO2 interaction, and performed model selection with an F-test. Our final model involved only a linear PCO2 term and an intercept.

When reporting your results, it's a good idea to report three aspects for each covariate. Firstly, you should always report the coefficient estimate. The coefficient estimate allows the reader to assess the magnitude of the effect. There are many circumstances where a result may be statistically significant, but practically meaningless. Secondly, alongside your estimate you should always report some measure of uncertainty or precision. For linear regression, the standard error (`Std. Error` column in the `R` output) can be reported. We will cover another method called a confidence interval later on in this section. Lastly, reporting a *p*-value for each of the coefficients is also a good idea. An example of appropriate presentation of our final model would be something similar to: TCO2 increased 0.18 (SE: 0.008, *p*-value <0.001) units per unit increase of PCO2. You will note we reported *p*-value <0.001, when in fact it is smaller than this. It is common to report very small *p*-values as <0.001 or <0.0001 instead of using a large number of decimal places. While sometimes it's simply reported whether $p < 0.05$ or not (i.e., if the result is statistically significant or not), this practice should be avoided.

Often it's a good idea to also discuss how well the overall model fit. There are several ways to accomplish this, but reporting a unitless quantity known as R^2 (pronounced r-squared) is often done. Looking back to the output `R` provided for our chosen final model, we can find the value of R^2 for this model under `Multiple R-squared`: 0.2647. This quantity is a proportion (a number between 0 and 1), and describes how much of the total variability in the data is explained by the model. An R^2 of 1 indicates a perfect fit, where 0 explains no variability in the data. What exactly constitutes a 'good' R^2 depends on subject matter and how it will be used. Another way to describe the fit in your model is through the residual standard error. This is also in the `lm` output when using the `summary` function. This roughly estimates square-root of the average squared distance between the model fit and the data. While it is in the same units as the outcome, it is in general more difficult to interpret than R^2. It should be noted that for evaluating prediction error, these values are likely too optimistic when applied to new data, and a better estimate of the error should be evaluated by other methods (e.g., cross-validation), which will be covered in another chapter and elsewhere [4, 5].

Interpreting the Results

Interpreting the results is an important component to any data analysis. We have already covered interpreting the intercept, which is the prediction for the outcome when all covariates are set at zero. This quantity is not of direct interest in most studies. If one does want to interpret it, subtracting the mean from each of the model's covariates will make it more interpretable—the expected value of the outcome when all covariates are set to the study's averages.

The coefficient estimates for the covariates are in general the quantities most of scientific interest. When the covariate is binary (e.g., `gender_num`), the coefficient represents the difference between one level of the covariate (1) relative to the other level (0), while holding any other covariates in the model constant. Although we won't cover it until the next section, extending discrete covariates to the case when they have more than two levels (e.g., ethnicity or `service_unit`) is quite similar, with the noted exception that it's important to reference the baseline group (i.e., what is the effect relative to). We will return to this topic later on in the chapter. Lastly, when the covariate is continuous the interpretation is the expected change in the outcome as a result of increasing the covariate in question by one unit, while holding all other covariates fixed. This interpretation is actually universal for any non-intercept coefficient, including for binary and other discrete data, but relies more heavily on understanding how `R` is coding these covariates with dummy variables.

We examined statistical interactions briefly, and this topic can be very difficult to interpret. It is often advisable, when possible, to represent the interaction graphically, as we did in Fig. 16.3.

Confidence and Prediction Intervals

As mentioned above, one method to quantify the uncertainty around coefficient estimates is by reporting the standard error. Another commonly used method is to report a confidence interval, most commonly a 95 % confidence interval. A 95 % confidence interval for β is an interval for which if the data were collected repeatedly, about 95 % of the *intervals* would contain the *true value* of the parameter, β, assuming the modeling assumptions are correct.

To get 95 % confidence intervals of coefficients, `R` has a `confint` function, which you pass an `lm` object to. It will then output 2.5 and 97.5 % confidence interval limits for each coefficient.

```
confint(co2.lm)
```

```
##                  2.5 %     97.5 %
## (Intercept) 15.5053693 16.9163494
## pco2_first   0.1731033  0.2040403
```

The 95 % confidence interval for `pco2_first` is about 0.17–0.20, which may be slightly more informative than reporting the standard error. Often people will look at if the confidence interval includes zero (no effect). Since it does not, and in fact since the interval is quite narrow and not very close to zero, this provides some additional evidence of its importance. There is a well known link between hypothesis testing and confidence intervals which we will not get into detail here.

When plotting the data with the model fit, similar to Fig. 16.2, it is a good idea to include some sort of assessment of uncertainty as well. To do this in R, we will first create a data frame with PCO2 levels which we would like to predict. In this case, we would like to predict the outcome (TCO2) over the range of observed covariate (PCO2) values. We do this by creating a data frame, where the variable names in the data frame must match the covariates used in the model. In our case, we have only one covariate (pco2_first), and we predict the outcome over the range of covariate values we observed determined by the min and max functions.

```
grid.pred <- data.frame(pco2_first=seq.int(from=min(dat$pco2_first,na.rm=T),
                                           to=max(dat$pco2_first,na.rm=T)));
```

Then, by using the predict function, we can predict TCO2 levels at these PCO2 values. The predict function has three arguments: the model we have constructed (in this case, using lm), newdata, and interval. The newdata argument allows you to pass any data frame with the same covariates as the model fit, which is why we created grid.pred above. Lastly, the interval argument is optional, and allows for the inclusion of any confidence or prediction intervals. We want to illustrate a prediction interval which incorporates both uncertainty about the model coefficients, in addition to the uncertainty generated by the data generating process, so we will pass interval = ”prediction”.

```
preds <- predict(co2.lm,newdata=grid.pred,interval = "prediction")
preds[1:2,]
```

```
##        fit      lwr      upr
## 1 17.71943 9.078647 26.36022
## 2 17.90801 9.268186 26.54783
```

We have printed out the first two rows of our predictions, preds, which are the model’s predictions for PCO2 at 8 and 9. We can see that our predictions (fit) are about 0.18 apart, which make sense given our estimate of the slope (0.18). We also see that our 95 % prediction intervals are very wide, spanning about 9 (lwr) to 26 (upr). This indicates that, despite coming up with a model which is very statistically significant, we still have a lot of uncertainty about the predictions generated from such a model. It is a good idea to capture this quality when plotting how well your model fits by adding the interval lines as dotted lines. Let’s plot our final model fit, co2.lm, along with the scatterplot and prediction interval in Fig. 16.4.

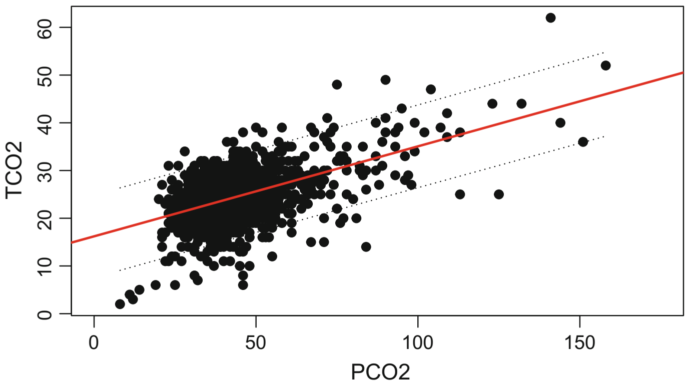


Fig. 16.4 Scatterplot of PCO2 (x-axis) and TCO2 (y-axis) along with linear regression estimates from the linear only model (co2.lm). The *dotted line* represents 95 % prediction intervals for the model

```
plot(dat$pco2_first,dat$tco2_first,xlab="PCO2",ylab="TCO2",pch=19,xlim=c(0,175))
co2.lm <- lm(tco2_first ~ pco2_first,data=dat)
abline(co2.lm,col='red',lwd=2)
lines(grid.pred$pco2_first,preds[,2],lty=3)
lines(grid.pred$pco2_first,preds[,3],lty=3)
```

16.2.5 Caveats and Conclusions

Linear regression is an extremely powerful tool for doing data analysis on continuous outcomes. Despite this, there are several aspects to be aware of when performing this type of analysis.

1. Hypothesis testing and the interval generation are reliant on modelling assumptions. Doing diagnostic plots is a critical component when conducting data analysis. There is subsequent discussion on this elsewhere in the book, and we will refer you to [6–8] for more information about this important topic.
2. Outliers can be problematic when fitting models. When there are outliers in the covariates, it's often easiest to turn a numeric variable into a categorical one (2 or more groups cut along values of the covariate). Removing outliers should be avoided when possible, as they often tell you a lot of information about the data generating process. In other cases, they may identify problems for the extraction process. For instance, a subset of the data may use different units for the same covariate (e.g., inches and centimeters for height), and thus the data needs to be converted to common units. Methods robust to outliers are available in R, a brief introduction of how to get started with some of the functions in R is available [7].

3. Be concerned about missing data. `R` reports information about missing data in the `summary` output. For our model fit `co2.lm`, we had 186 observations with missing `pco2_first` observations. `R` will leave these observations out of the analysis, and fit on the remaining non-missing observations. Always check the output to ensure you have as many observations as you think that you are supposed to. When many observations have missing data and you try to build a model with a large number of coefficients, you may be fitting the model on only a handful of observations.
4. Assess potential multi-colinearity. Co-linearity can occur when two or more covariates are highly correlated. For instance, if blood pressure on the left and right arms were simultaneously measured, and both used as covariates in the model. In this case, consider taking the sum, average or difference (whichever is most useful in the particular case) to craft a single covariate. Co-linearity can also occur when a categorical variable has been improperly generated. For instance, defining groups along the PCO2 covariate of 0–25, 5–26, 26–50, >50 may cause linear regression to encounter some difficulties as the first and second groups are nearly identical (usually these types of situations are programming errors). Identifying covariates which may be colinear is a key part of the exploratory analysis stage, where they can often (but not always) be seen by plotting the data.
5. Check to see if outcomes are dependent. This most commonly occurs when one patient contributes multiple observations (outcomes). There are alternative methods for dealing with this situation [9], but it is beyond the scope of this chapter.

These concerns should not discourage you from using linear regression. It is extremely powerful and reasonably robust to some of the problems discussed above, depending on the situation. Frequently a continuous outcome is converted to a binary outcome, and often there is no compelling reason this is done. By discretizing the outcome you may be losing information about which patients may benefit or be harmed most by a therapy, since a binary outcome may treat patients who had very different outcomes on the continuous scale as the same. The overall framework we took in linear regression will closely mirror the way in which we approach the other analysis techniques we discuss later in this chapter.

16.3 Logistic Regression

16.3.1 Section Goals

In this section, the reader will learn the fundamentals of logistic regression, and how to present and interpret such an analysis.

16.3.2 Introduction

In Sect. 16.2 we covered a very useful methodology for modeling quantitative or continuous outcomes. We of course know though that health outcomes come in all different kinds of data types. In fact, the health outcomes we often care about most —cured/not cured, alive/dead, are discrete binary outcomes. It would be ideal if we could extend the same general framework for continuous outcomes to these binary outcomes. Logistic regression allows us to incorporate much of what we learned in the previous section and apply the same principles to binary outcomes.

When dealing with binary data, we would like to be able to model the probability of a type of outcome given one or more covariates. One might ask, why not just simply use linear regression? There are several reasons why this is generally a bad idea. Probabilities need to be somewhere between zero and one, and there is nothing in linear regression to constrain the estimated probabilities to this interval. This would mean that you could have an estimated probability 2, or even a negative probability! This is one unattractive property of such a method (there are others), and although it is sometimes used, the availability of good software such as R allows us to perform better analyses easily and efficiently. Before introducing such software, we should introduce the analysis of small contingency tables.

16.3.3 2 × 2 Tables

Contingency tables are the best way to start to think about binary data. A contingency table cross-tabulates the outcome across two or more levels of a covariate. Let's begin by creating a new variable (age.cat) which dichotomizes age into two age categories: ≤ 55 and > 55. Note, because we are making age a discrete variable, we also change the data type to a factor. This is similar to what we did for the gender_num variable when discussing linear regression in the previous section. We can get a breakdown of the new variable using the table function.

```
dat$age.cat <- as.factor(ifelse(dat$age<=55, "<=55",">55"))
table(dat$age.cat)
```

```
##
## <=55  >55
##  923  853
```

We would like to see how 28 day mortality is distributed among the age categories. We can do so by constructing a contingency table, or in this case what is commonly referred to as a 2 × 2 table.

```
table(dat$age.cat,dat$day_28_flg)
```

```
##
##          0   1
##   <=55 883  40
##   >55  610 243
```

From the above table, you can see that 40 patients in the young group (≤ 55) died within 28 days, while 243 in the older group died. These correspond to $P(\text{die}|\text{age} \leq 55) = 0.043)$ or 4.3 % and $P(\text{die}|\text{age} > 55) = 0.284$ or 28.4 %, where the "|" can be interpreted as "given" or "for those who have." This difference is quite marked, and we know that age is an important factor in mortality, so this is not surprising.

The odds of an event happening is a positive number and can be calculated from the probability of an event, p, by the following formula

$$\text{Odds} = \frac{p}{1-p}.$$

An event with an odds of zero never happens, and an event with a very large odds (>100) is very likely to happen. Here, the odds of dying within 28 days in the young group is 0.043/(1 − 0.043) = 0.045, and in the older group is 0.284/(1 −0.284) = 0.40. It is convenient to represent these two figures as a ratio, and the choice of what goes in the numerator and the denominator is somewhat arbitrary. In this case, we will choose to put the older group's odds on the numerator and the younger in the denominator, and it's important to make it clear which group is in the numerator and denominator in general. In this case the *Odds ratio* is 0.40/0.045 = 8.79, which indicates a very strong association between age and death, and means that the odds of dying in the older group is nearly 9 fold higher than when compared to the younger group. There is a convenient shortcut for doing odds ratio calculation by making an X on a 2 × 2 table and multiplying top left by bottom right, then dividing it by the product of bottom left and top right. In this case $\frac{883 \times 243}{610 \times 40} = 8.79$.

Now let us look at a slightly different case—when the covariate takes on more than two values. Such a variable is the `service_unit`. Let's see how the deaths are distributed among the different units:

```
deathbyservice <- table(dat$service_unit,dat$day_28_flg)
deathbyservice
```

```
##
##          0   1
##   FICU  59   3
##   MICU 605 127
##   SICU 829 153
```

we can get frequencies of these service units by applying the `prop.table` function to our cross-tabulated table.

```
dbys.proptable <- prop.table(deathbyservice,1)
dbys.proptable
```

```
##
##                0         1
##   FICU 0.9516129 0.0483871
##   MICU 0.8265027 0.1734973
##   SICU 0.8441955 0.1558045
```

It appears as though the `FICU` may have a lower rate of death than either the `MICU` or `SICU`. To compute an odds ratios, first compute the odds:

```
dbys.proptable[,"1"]/dbys.proptable[,"0"]
```

```
##       FICU       MICU       SICU
## 0.05084746 0.20991736 0.18455971
```

and then we need to pick which of `FICU`, `MICU` or `SICU` will serve as the reference or baseline group. This is the group which the other two groups will be compared to. Again the choice is arbitrary, but should be dictated by the study objective. If this were a clinical trial with two drug arms and a placebo arm, it would be foolish to use one of the treatments as the reference group, particularly if you wanted to compare the efficacy of the treatments. In this particular case, there is no clear reference group, but since the FICU is so much smaller than the other two units, we will use it as the reference group. Computing the odds ratio for MICU and SICU we get 4.13 and 3.63, respectively. These are also very strong associations, meaning that the odds of dying in the SICU and MICU are around 4 times higher than in the FICU, but relatively similar.

Contingency tables and 2×2 tables in particular are the building blocks of working with binary data, and it's often a good way to begin looking at the data.

16.3.4 Introducing Logistic Regression

While contingency tables are a fundamental way of looking at binary data, they are somewhat limited. What happens when the covariate of interest is continuous? We could of course create categories from the covariate by establishing cut points, but we may still miss some important aspect of the relationship between the covariate and the outcome by not choosing the right cut points. Also, what happens when we know that a nuisance covariate is related to both the outcome and the covariate of interest. This type of nuisance variable is called a confounder and occurs frequently in observational data, and although there are ways of accounting for confounding in contingency tables, they become more difficult to use when there are more than one present.

Logistic regression is a way of addressing both of these issues, among many others. If you recall, using linear regression is problematic because it is prone to estimating probabilities outside of the [0, 1] range. Logistic regression has no such problem per se, because it uses a link function known as the logit function which maps probabilities in the interval $[0,1]$ to a real number $(-\infty,\infty)$. This is important for many practical and technical reasons. The logit of p_x (i.e. the probability of an event for certain covariate values x) is related to the covariates in the following way

$$\mathrm{logit}(p_x) = \log(Odds_x) = \log(\frac{p_x}{1-p_x}) = \beta_0 + \beta_1 \times x.$$

It is worth pointing out here that log here, and in most places in statistics is referring to the natural logarithm, sometimes denoted *ln*.

The first covariate we were considering, `age.cat` was also a binary variable, where it takes on values 1 when the `age` > 55 and 0 when `age` ≤ 55. So plugging these values in, first for the young group $(x = 0)$:

$$\mathrm{logit}(p_{x=0}) = \log(Odds_{x=0}) = \log(\frac{p_{x=0}}{1-p_{x=0}}) = \beta_0 + \beta_1 \times 0 = \beta_0,$$

and then for the older group $(x = 1)$:

$$\mathrm{logit}(p_{x=1}) = \log(Odds_{x=1}) = \log(\frac{p_{x=1}}{1-p_{x=1}}) = \beta_0 + \beta_1 \times 1 = \beta_0 + \beta_1.$$

If we subtract the two cases $\mathrm{logit}(p_{x=1}) - \mathrm{logit}(p_{x=0}) = \log(Odds_{x=1}) - \log(Odds_{x=0})$, and we notice that this quantity is equal to β_1. If you recall the properties of logarithms, that the difference of two logs is the log of their ratio, so $\log(Odds_{x=1}) - \log(Odds_{x=0}) = \log(Odds_{x=1}/Odds_{x=0})$, which may be looking familiar. This is the log ratio of the odds or the *log odds ratio* in the $x = 1$ group relative to the $x = 0$ group. Hence, we can estimate odds ratios using logistic regression by exponentiating the coefficients of the model (the intercept notwithstanding, which we will get to in a moment).

Let's fit this model, and see how this works using a real example. We fit logistic regression very similarly to how we fit linear regression models, with a few exceptions. First, we will use a new function called `glm`, which is a very powerful function in `R` which allow one to fit a class of models known as generalized linear models or GLMs [10]. The `glm` function works in much the same way the `lm` function does. We need to specify a formula of the form: `outcome` $\sim$ `covariates`, specify what dataset to use (in our case the `dat` data frame), and then specify the family. For logistic regression `family` = '`binomial`' will be our choice. You can run the `summary` function, just like you did for `lm` and it produces output very similar to what `lm` did.

```
age.glm <- glm(day_28_flg ~ age.cat,data=dat,family="binomial")
summary(age.glm)
```

```
##
## Call:
## glm(formula = day_28_flg ~ age.cat, family = "binomial", data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8189  -0.8189  -0.2977  -0.2977   2.5055
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.0944     0.1616  -19.14   <2e-16 ***
## age.cat>55    2.1740     0.1785   12.18   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1557.9  on 1775  degrees of freedom
## Residual deviance: 1348.7  on 1774  degrees of freedom
## AIC: 1352.7
##
## Number of Fisher Scoring iterations: 5
```

As you can see, we get a coefficients table that is similar to the `lm` table we used earlier. Instead of a `t value`, we get a `z value`, but this can be interpreted similarly. The rightmost column is a p-value, for testing the null hypothesis $\beta = 0$. If you recall, the non-intercept coefficients are log-odds ratios, so testing if they are zero is equivalent to testing if the odds ratios are one. If an odds ratio is one the odds are equal in the numerator group and denominator group, indicating the probabilities of the outcome are equal in each group. So, assessing if the coefficients are zero will be an important aspect of doing this type of analysis.

Looking more closely at the coefficients. The intercept is −3.09 and the `age.cat` coefficient is 2.17. The coefficient for `age.cat` is the log odds ratio for the 2 × 2 table we previously did the analysis on. When we exponentiate 2.17, we get `exp`(2.17) = 8.79. This corresponds with the estimate using the 2 × 2 table. For completeness, let's look at the other coefficient, the intercept. If you recall, $\log(Odds_{x=0}) = \beta_0$, so β_0 is the log odds of the outcome in the younger group. Exponentiating again, `exp`(−3.09) = 0.045, and this corresponds with the previous analysis we did. Similarly, $\log(Odds_{x=1}) = \beta_0 + \beta_1$, and the estimated odds of 28 day death in the older group is `exp`(−3.09 + 2.17) = 0.4, as was found above. Converting estimated odds into a probability can be done directly using the `plogis` function, but we will cover a more powerful and easier way of doing this later on in the section.

Beyond a Single Binary Covariate

While the above analysis is useful for illustration, it does not readily demonstrate anything we could not do with our 2 × 2 table example above. Logistic regression allows us to extend the basic idea to at least two very relevant areas. The first is the case where we have more than one covariate of interest. Perhaps we have a confounder, we are concerned about, and want to adjust for it. Alternatively, maybe there are two covariates of interest. Secondly, it allows use to use covariates as continuous quantities, instead of discretizing them into categories. For example, instead of dividing age up into exhaustive strata (as we did very simply by just dividing the patients into two groups, ≤ 55 and > 55), we could instead use age as a continuous covariate.

First, having more than one covariate is simple. For example, if we wanted to add service_unit to our previous model, we could just add it as we did when using the lm function for linear regression. Here we specify ∼day_28_flg age.cat + service_unit and run the summary function.

```
ageunit.glm <- glm(day_28_flg ~ age.cat + service_unit,data=dat,family="binomial")
summary(ageunit.glm)$coef
```

```
##                    Estimate Std. Error   z value     Pr(>|z|)
## (Intercept)       -4.209013  0.6222758 -6.763903 1.343230e-11
## age.cat>55         2.161142  0.1787575 12.089800 1.195779e-33
## service_unitMICU  1.178865  0.6151757  1.916307 5.532607e-02
## service_unitSICU  1.123442  0.6135095  1.831173 6.707466e-02
```

A coefficient table is produced, and now we have four estimated coefficients. The same two, (Intercept) and age.cat which were estimated in the unadjusted model, but also we have service_unitMICU and service_unitSICU which correspond to the log odds ratios for the MICU and SICU relative to the FICU. Taking the exponential of these will result in an odds ratio for each variable, adjusted for the other variables in the model. In this case the adjusted odds ratios for Age > 55, MICU and SICU are 8.68, 3.25, and 3.08, respectively. We would conclude that there is an almost 9-fold increase in the odds of 28 day mortality for those in the >55 year age group relative to the younger ≤ 55 group while holding service unit constant. This adjustment becomes important in many scenarios where groups of patients may be more or less likely to receive treatment, but also more or less likely to have better outcomes, where one effect is confounded by possibly many others. Such is almost always the case with observational data, and this is why logistic regression is such a powerful data analysis tool in this setting.

Another case we would like to be able to deal with is when we have a continuous covariate we would like to include in the model. One can always break the continuous covariate into mutually exclusive categories by selecting break or cut points, but selecting the number and location of these points can be arbitrary, and in many cases unnecessary or inefficient. Recall that in logistic regression we are fitting a model:

$$\text{logit}(p_x) = \log(Odds_x) = \log(\frac{p_x}{1-p_x}) = \beta_0 + \beta_1 \times x,$$

but now assume x is continuous. Imagine a hypothetical scenario where you know β_0 and β_1 and have a group of 50 year olds, and a group of 51 year olds. The difference in the log Odds between the two groups is:

$$\begin{aligned}\log(Odds_{51}) - \log(Odds_{50}) = (\beta_0 + \beta_1 \times 51) - (\beta_0 + \beta_1 \times 50) = \beta_1(51-50)\\ = \beta_1.\end{aligned}$$

Hence, the odds ratio for 51 year olds versus 50 year olds is $\exp(\beta_1)$. This is actually true for any group of patients which are 1 year apart, and this gives a useful way to interpret and use these estimated coefficients for continuous covariates. Let's work with an example. Again fitting the 28 day mortality outcome as a function of age, but treating age as it was originally recorded in the dataset, a continuous variable called age.

```
agects.glm <- glm(day_28_flg ~ age,data=dat,family="binomial")
summary(agects.glm)$coef
```

```
##                Estimate  Std. Error   z value     Pr(>|z|)
## (Intercept) -5.77800634 0.320774776 -18.01266 1.550034e-72
## age          0.06523274 0.004469569  14.59486 3.028256e-48
```

We see the estimated coefficient is 0.07 and still very statistically significant. Exponentiating the log odds ratio for age, we get an estimated odds ratio of 1.07, which is per 1 year increase in age. What if the age difference of interest is ten years instead of one year? There are at least two ways of doing this. One is to replace age with I(age/10), which uses a new covariate which is age divided by ten. The second is to use the agects.glm estimated log odds ratio, and multiple by ten prior to exponentiating. They will yield equivalent estimates of 1.92, but it is now per 10 year increases in age. This is useful when the estimated odds ratios (or log odds ratios) are close to one (or zero). When this is done, one unit of the covariate is 10 years, so the generic interpretation of the coefficients remains the same, but the units (per 10 years instead of per 1 year) changes.

This of course assumes that the form of our equation relating the log odds of the outcome to the covariate is correct. In cases where odds of the outcome decreases and increases as a function of the covariate, it is possible to estimate a relatively small effect of the linear covariate, when the outcome may be strongly affected by the covariate, but not in the way the model is specified. Assessing the linearity of the log odds of the outcome and some discretized form of the covariate can be done graphically. For instance, we can break age into 5 groups, and estimate the log odds of 28 day mortality in each group. Plotting these quantities in Fig. 16.5 (left), we can see in this particular case, age is indeed strongly related to the odds of the outcome. Further, expressing age linearly appears like it would be a good approximation. If on the other hand, 28 day mortality has more of a “U”-shaped curve, we may falsely conclude that no relationship between age and mortality exists, when the relationship may be rather strong. Such may be the case when looking at the the log odds of mortality by the first temperature (temp_1st) in Fig. 16.5 (right).

16.3.5 Hypothesis Testing and Model Selection

Just as in the case for linear regression, there is a way to test hypotheses for logistic regression. It follows much of the same framework, with the null hypothesis being $\beta = 0$. If you recall, this is the log odds ratio, and testing if it is zero is equivalent to a test for the odds ratio being equal to one. In this chapter, we focus on how to conduct such a test in R.

As was the case when using lm, we first fit the two competing models, a larger (alternative model), and a smaller (null model). Provided that the models are nested, we can again use the anova function, passing the smaller model, then the larger model. Here our larger model is the one which contained service_unit and age.cat, and the smaller only contains age.cat, so they are nested. We are then testing if the log odds ratios for the two coefficients associated with service_unit are zero. Let’s call these coefficients β_{MICU} and β_{SICU}. To test if β_{MICU} and $\beta_{SICU} = 0$, we can use the anova function, where this time we will specify the type of test, in this case set the test parameter to “Chisq”.

```r
anova(age.glm,ageunit.glm,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: day_28_flg ~ age.cat
## Model 2: day_28_flg ~ age.cat + service_unit
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      1774     1348.7
## 2      1772     1343.8  2   4.9315  0.08495 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here the output of the anova function when applied to glm objects looks similar to the output generated when used on lm objects. A couple good practices to get in a habit are to first make sure the two competing models are correctly specified. He we are are testing $\sim$ age.cat versus age.cat + service_unit. Next, the difference between the residual degrees of freedom (Resid. Df) in the two models tell us how many more parameters the larger model has when compared

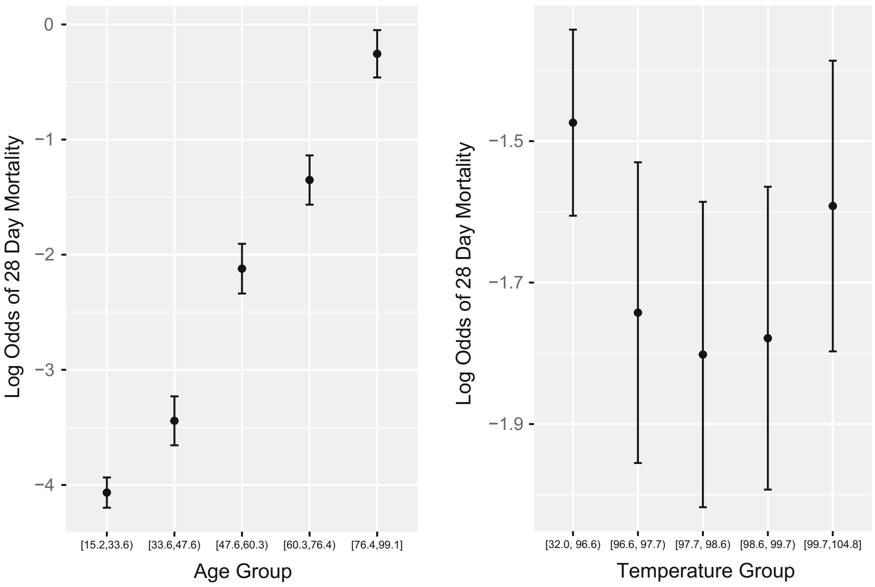


Fig. 16.5 Plot of log-odds of mortality for each of the five age and temperature groups. *Error bars* represent 95 % confidence intervals for the log odds

to the smaller model. Here we see `1774 − 1772 = 2` which means that there are two more coefficients estimated in the larger model than the smaller one, which corresponds with the output from the `summary` table above. Next looking at the *p*-value (`Pr(>Chi)`), we see a test for β_{MICU} and $\beta_{SICU} = 0$ has a *p*-value of around 0.08. At the typical 0.05 significance level, we would not reject the null, and use the simpler model without the service unit. In logistic regression, this is a common way of testing whether a categorical covariate should be retained in the model, as it can be difficult to assess using the `z value` in the `summary` table, particularly when one is very statistically significant, and one is not.

16.3.6 Confidence Intervals

Generating confidence intervals for either the log-odds ratios or the odds ratios are relatively straightforward. To get the log-odds ratios and respective confidence intervals for the `ageunit.glm` model which includes both age and service unit.

```
ageunit.glm$coef
```

```
##      (Intercept)       age.cat>55 service_unitMICU service_unitSICU
##        -4.209013         2.161142         1.178865         1.123442
```

```
confint(ageunit.glm)
```

```
##                        2.5 %    97.5 %
## (Intercept)      -5.66202924 -3.139732
## age.cat>55        1.82211403  2.524682
## service_unitMICU  0.12291680  2.620797
## service_unitSICU  0.07182767  2.563132
```

Here the coefficient estimates and confidence intervals are presented in much the same way as for a linear regression. In logistic regression, it is often convenient to exponentiate these quantities to get it on a more interpretable scale.

```
exp(ageunit.glm$coef[-1])
```

```
##       age.cat>55 service_unitMICU service_unitSICU
##         8.681049         3.250684         3.075423
```

```
exp(confint(ageunit.glm)[-1,])
```

```
##                    2.5 %   97.5 %
## age.cat>55       6.18492 12.48693
## service_unitMICU 1.13079 13.74668
## service_unitSICU 1.07447 12.97640
```

Similar to linear regression, we will look at if the confidence intervals for the log odds ratios include zero. This is equivalent to seeing if the intervals for the odds ratios include 1. Since the odds ratios are more directly interpretable it is often more convenient to report them instead of the coefficients on the log odds ratio scale.

16.3.7 Prediction

Once you have decided on your final model, you may want to generate predictions from your model. Such a task may occur when doing a propensity score analysis (Chap. 25) or creating tools for clinical decision support. In the logistic regression setting this involves attempting to estimate the probability of the outcome given the characteristics (covariates) of a patient. This quantity is often denoted $P(outcome|X)$. This is relatively easy to accomplish in `R` using the `predict` function. One must pass a dataset with all the variables contained in the model. Let's assume that we decided to include the `service_unit` in our final model, and want to generate predictions from this based on a new set of patients. Let's first create a new data frame called newdat using the expand.grid function which computes all combinations of the values of variables passed to it.

```
newdat <- expand.grid(age.cat=c("<=55",">55"),service_unit=c("FICU","MICU","SICU"))
newdat$pred <- predict(ageunit.glm,newdata=newdat,type="response")
newdat
```

```
##   age.cat service_unit       pred
## 1    <=55         FICU 0.01464341
## 2     >55         FICU 0.11426771
## 3    <=55         MICU 0.04608233
## 4     >55         MICU 0.29546130
## 5    <=55         SICU 0.04370639
## 6     >55         SICU 0.28405645
```

We followed this by adding a pred column to our new data frame by using the predict function. The predict function for logistic regression works similar to when we used it for linear regression, but this time we also specify type = ”response” which ensures the quantities computed are what we need, $P(outcome|X)$. Outputting this new object shows our predicted probability of 28 day mortality for six hypothetical patients. Two in each of the service units, where one is in the younger group and another in the older group. We see that our lowest prediction is for the youngest patients in the FICU, while the patients with highest risk of 28 day mortality are the older group in the MICU, but the predicted probability is not all that much higher than the same age patients in the SICU.

To do predictions on a different dataset, just replace the newdata argument with the other dataset. We could, for instance, pass newdata = dat and receive predictions for the dataset we built the model on. As was the case with linear regression, evaluating the predictive performance of our model on data used to build the model will generally be too optimistic as to how well it would perform *in the real world*. How to get a better sense of the accuracy of such models is covered in Chap. 17.

16.3.8 Presenting and Interpreting Logistic Regression Analysis

In general, presenting the results from a logistic regression model will follow quite closely to what was done in the linear regression setting. Results should always be put in context, including what variables were considered and which variables were in the final model. Reporting the results should always include some form of the coefficient estimate, a measure of uncertainty and likely a *p*-value. In medical and epidemiological journals, coefficients are usually exponentiated so that they are no longer on the log scale, and reported as odds ratios. Frequently, multivariable analyses (analysis with more than one covariate) is distinguished from univariate analyses (one covariate) by denoting the estimated odds ratios as adjusted odds ratios (AOR).

For the `age.glm` model, an example of what could be reported is:

> Mortality at 28 days was much higher in the older ($>$ 55 years) group than the younger group ($\leq$ 55 years), with rates of 28.5 and 4.3 %, respectively (OR = 8.79, 95 % CI: 6.27-12.64, p < 0.001).

When treating age as a continuous covariate in the `agects.glm` model we could report:

> Mortality at 28 days was associated with older age (OR = 1.07 per year increase, 95 % CI: 1.06–1.08, p < 0.001).

And for the case with more than one covariate, (`ageunit.glm`) an example of what could be reported:

> Older age ($>$ 55 versus $\leq$ 55 years) was independently associated with 28 day mortality (AOR = 8.68, 95 % CI: 6.18-12.49, p < 0.001) after adjusting for service unit.

16.3.9 Caveats and Conclusions

As was the case with linear regression, logistic regression is an extremely powerful tool for data analysis of health data. Although the study outcomes in each approach are different, the framework and way of thinking of the problem have similarities. Likewise, many of the problems encountered in linear regression are also of concern in logistic regression. Outliers, missing data, colinearity and dependent/correlated outcomes are all problems for logistic regression as well, and can be dealt with in a similar fashion. Modelling assumptions are as well, and we briefly touched on this when discussing whether it was appropriate to use age as a continuous covariate in our models. Although continuous covariates are frequently modeled in this way, it is important to ensure if the relationship between the log odds of the outcome is indeed linear with the covariate. In cases where the data has been divided into too many subgroups (or the study may be simply too small), you may encounter a level of a discrete variable where none (or very few) of one of the outcomes occurred. For example, if we had an additional `service_unit` with 50 patients, all of whom lived. In such a case, the estimated odds ratios and subsequent confidence intervals or hypothesis testing may not be appropriate to use. In such a case, collapsing the discrete covariate into fewer categories will often help return the analysis into a manageable form. For our hypothetical new service unit, creating a new group of it and FICU would be a possible solution. Sometimes a covariate is so strongly related to the outcome, and this is no longer possible, and the only solution may be to report this finding, and remove these patients.

Overall, logistic regression is a very valuable tool in modelling binary and categorical data. Although we did not cover this latter case, a similar framework is available for discrete data which is ordered or has more than one category (see `?multinom` in the `nnet` package in `R` for details about multinomial logistic regression). This and other topics such as assessing model fit, and using logistic regression in more complicated study designs are discussed in [11].

16.4 Survival Analysis

16.4.1 Section Goals

In this section, the reader will learn the fundamentals of survival analysis, and how to present and interpret such an analysis.

16.4.2 Introduction

As you will note that in the previous section on logistic regression, we specifically looked at the mortality outcome at 28 days. This was deliberate, and illustrates a limitation of using logistic regression for this type of outcome. For example, in the previous analysis, someone who died on day 29 was treated identically as someone who went on to live for 80+ years. You may wonder, why not just simply treat the survival time as a continuous variable, and perform linear regression analysis on this outcome? There are several reasons, but the primary reason is that you likely won't be able to wait around for the lifetime for each study participant. It is likely in your study only a fraction of your subjects will die before you're ready to publish your results.

While we often focus on mortality this can occur for many other outcomes, including times to patient relapse, re-hospitalization, reinfection, etc. In each of these types of outcomes, it is presumed the patients are at risk of the outcome until the event happens, or until they are *censored*. Censoring can happen for a variety of different reasons, but indicates the event was not observed during the observation time. In this sense, survival or more generally time-to-event data is a bivariate outcome incorporating the observation or study time in which the patient was observed and whether the event happened during the period of observation. The particular case we will be most interested is *right censoring* (subjects are observed only up to a point in time, and we don't know what happens beyond this point), but there is also *left censoring* (we only know the event happened before some time point) and *interval censoring* (events happen inside some time window). Right censoring is generally the most common type, but it is important to understand how the data was collected to make sure that it is indeed right censored.

Establishing a common time origin (i.e., a place to start counting time) is often easy to identify (e.g., admission to the ICU, enrollment in a study, administration of a drug, etc.), but in other scenarios it may not be (e.g., perhaps interest lies in survival time since disease onset, but patients are only followed from the time of disease diagnosis). For a good treatment on this topic and other issues, see Chap. 3 of [12].

With this additional complexity in the data (relative to logistic and linear regression), there are additional technical aspects and assumptions to the data analysis approaches. In general, each approach attempts to compare groups or identify covariates which modify the survival rates among the patients studied.

Overall survival analysis is a complex and fascinating area of study, and we will only touch briefly on two types of analysis here. We largely ignore the technical details of these approaches focusing on general principles and intuition instead. Before we begin doing any survival analysis, we need to load the `survival` package in `R`, which we can do by running:

```
library(survival);
```

Normally, you can skip the next step, but since this dataset was used to analyze the data in a slightly different way, we need to correct the observation times for a subset of the subjects in the dataset.

```
dat$mort_day_censored[dat$censor_flg==1] <- 731;
```

16.4.3 Kaplan-Meier Survival Curves

Now that we have the technical issues sorted out, we can begin by visualizing the data. Just as the 2 × 2 table is a fundamental step in the analysis of binary data, the fundamental step for survival data is often plotting what is known as a Kaplan-Meier survival function [13]. The *survival function* is a function of time, and is the probability of surviving at least that amount of time. For example, if there was 80 % survival at one year, the survival function at one year is 0.8. Survival functions normally start at `time = 0`, where the survivor function is 1 (or 100 % – everyone is alive), and can only stay the same or decrease. If it were to increase as time progressed, that would mean people were coming back to life! Kaplan-Meier plots are one of the most widely used plots in medical research.

Before plotting the Kaplan-Meier plot, we need to setup a `survfit` object. This object has a familiar form, but differs slightly from the previous methodologies we covered. Specifying a formula for survival outcomes is somewhat more complicated, since as we noted, survival data has two components. We do this by creating a `Surv` object in R. This will be our survival outcome for subsequent analysis.

```
datSurv <- Surv(dat$mort_day_censored,dat$censor_flg==0)
datSurv[101:105]
```

```
## [1] 236.08  731.00+ 731.00+ 731.00+   2.00
```

The first step setups a new kind of `R` object useful for survival data. The `Surv` function normally takes two arguments: a vector of times, and some kind of indicator for which patients had an event (death in our case). In our case, the vector of death and censoring times are the `mort_day_censored`, and deaths are coded with a zero in the `censor_flg` variable (hence we identify the events where `censor_flg == 0`). The last step prints out 5 entries of the new object (observations 101 to 105). We can see there are three entries of `731.00+`. The + indicates that this observation is censored. The other entries are not censored, indicating deaths at those times.

Fitting a Kaplan-Meier curve is quite easy after doing this, but requires two steps. The first specifies a formula similar to how we accomplished this for linear and logistic regression, but now using the `survfit` function. We want to ‘fit’ by gender (`gender_num`), so the formula is, `datSurv` ∼ `gender_num`. We can then `plot` the newly created object, but we pass some additional arguments to the plot function which include 95 % confidence intervals for the survival functions (`conf.int = TRUE`), and includes a x- and y- axis label (`xlab` and `ylab`). Lastly we add a legend, coding black for the women and red for the men. This plot is in Fig. 16.6.

```
gender.surv <- survfit(datSurv~gender_num,data=dat)
plot(gender.surv,col=1:2,conf.int = TRUE,xlab="Days",ylab="Proportion Who Survived")
legend(400,0.4,col=c("black","red"),lty=1,c("Women","Men"))
```

In Fig. 16.6, there appears to be a difference between the survival function between the two gender groups, with again the male group (red) dying at slightly slower rate than the female group (black). We have included 95 % point-wise confidence bands for the survival function estimate, which assesses how much certain we are about the estimated survivorship at each point in time. We can do the same for `service_unit`, but since it has three groups, we need to change the color argument and legend to ensure the plot is properly labelled. This plot is in Fig. 16.7.

```
unit.surv <- survfit(datSurv~service_unit,data=dat)
plot(unit.surv,col=1:3,conf.int = FALSE,xlab="Days",ylab="Proportion Who Survived")
legend(400,0.4,col=c("black","red","green"),lty=1,c("FICU","MICU","SICU"))
```

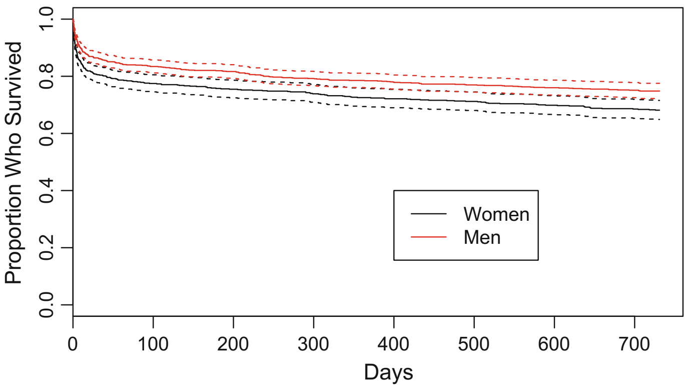


Fig. 16.6 Kaplan-Meier plot of the estimated survivor function stratified by gender

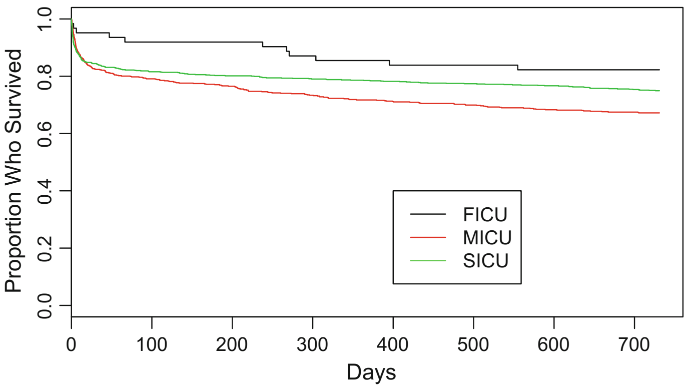


Fig. 16.7 Kaplan-Meier plot of the estimated survivor function stratified by service unit

16.4.4 Cox Proportional Hazards Models

Kaplan-Meier curves are a good first step in examining time to event data before proceeding with any more complex statistical model. Time to event outcomes are in general more complex than the other types of outcomes we have examined thus far. There are several different modelling approaches, each of which has some advantages and limitations. The most popular approach for health data is likely the Cox Proportional Hazards Model [14], which is also sometimes called the Cox model or Cox Regression. As the name implies this method models something called the hazard function. We will not dwell on the technical details, but attempt to provide some intuition. The hazard function is a function of time (hours, days, years) and is approximately the instantaneous probability of the event occurring (i.e., chance the event is happening in some very small time window) given the event has not already happened. It is frequently used to study mortality, sometimes going by the name force of mortality or instantaneous death rate, and can be interpreted simply as the risk of death at a particular time, given that the person has survived up until that point. The "proportional" part of Cox's model assumes that the way covariates effect the hazard function for different types of patients is through a proportionality assumption relative to the baseline hazard function. For illustration, consider a simple case where two treatments are given, for treatment 0 (e.g., the placebo) we determine the hazard function is $h_0(t)$, and for treatment 1 we determine the hazard function is $h_1(t)$, where t is time. The proportional hazards assumption is that:

$$h_1(t) = HR \times h_0(t).$$

It's easy to see that $HR = h_1(t)/h_0(t)$. This quantity is often called the hazard ratio, and if for example it is two, this would mean that the risk of death in the treatment 1 group was twice as high as the risk of death in the treatment zero group. We will note, that *HR* is *not* a function of time, meaning that the risk of death is *always* twice as high in the first group when compared to the second group. This assumption means that if the proportional hazards assumption is valid we need only know the hazard function from group 0, and the hazard ratio to know the hazard function for group 1. Estimation of the hazard function under this model is often considered a nuisance, as the primary focus is on the hazard ratio, and this is key to being able to fit and interpret these models. For a more technical treatment of this topic, we refer you to [12, 15–17].

As was the case with logistic regression, we will model the log of the hazard ratio instead of the hazard ratio itself. This allows us to use the familiar framework we have used thus far for modeling other types of health data. Like logistic regression, when the $\log(HR)$ is zero, the *HR* is one, meaning the risk between the groups is the same. Furthermore, this extends to multiple covariate models or continuous covariates in the same manner as logistic regression.

Fitting Cox regression models in `R` will follow the familiar pattern we have seen in the previous cases of linear and logistic regressions. The `coxph` function (from the `survival` package) is the fitting function for Cox models, and it continues the general pattern of passing a model formula (`outcome ~ covariate`), and the dataset you would like to use. In our case, let's continue our example of using gender (`gender_num`) to model the `datSurv` outcome we created, and running the `summary` function to see what information is outputted.

```
gender.coxph <- coxph(datSurv ~ gender_num,data=dat)
summary(gender.coxph)
```

```
## Call:
## coxph(formula = datSurv ~ gender_num, data = dat)
##
##   n= 1775, number of events= 497
##    (1 observation deleted due to missingness)
##
##                coef exp(coef) se(coef)     z Pr(>|z|)
## gender_num -0.29094   0.74756  0.08978 -3.24  0.00119 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##            exp(coef) exp(-coef) lower .95 upper .95
## gender_num    0.7476      1.338    0.6269    0.8914
##
## Concordance= 0.537  (se = 0.011 )
## Rsquare= 0.006   (max possible= 0.983 )
## Likelihood ratio test= 10.43  on 1 df,   p=0.001243
## Wald test            = 10.5  on 1 df,   p=0.001193
## Score (logrank) test = 10.58  on 1 df,   p=0.001146
```

The coefficients table has the familiar format, which we've seen before. The `coef` for `gender_num` is about −0.29, and this is the estimate of our log-hazard ratio. As discussed, taking the exponential of this gives the hazard ratio (HR), which the summary output computes in the next column (`exp(coef)`). Here, the HR is estimated at 0.75, indicating that men have about a 25 % reduction in the hazards of death, under the proportional hazards assumption.

The next column in the coefficient table has the standard error for the log hazard ratio, followed by the `z` score and *p*-value (`Pr(>|z|)`), which is very similar to what we saw in the case of logistic regression. Here we see the *p*-value is quite small, and we would reject the null hypothesis that the hazard functions are the same between men and women. This is consistent with the exploratory figures we produced using Kaplan-Meier curves in the previous section. For `coxph`, the `summary` function also conveniently outputs the confidence interval of the HR a few lines down, and here our estimate of the HR is 0.75 (95 % CI: 0.63–0.89, p = 0.001). This is how the HR would typically be reported.

Using more than one covariate works the same as our other analysis techniques. Adding a co-morbidity to the model such as atrial fibrillation (afib_flg) can be done as you would do for logistic regression.

```
genderafib.coxph <- coxph(datSurv~gender_num + afib_flg,data=dat)
summary(genderafib.coxph)$coef
```

```
##                  coef exp(coef)   se(coef)         z    Pr(>|z|)
## gender_num -0.2591201 0.7717304 0.08987143 -2.883231 0.003936189
## afib_flg    1.3443975 3.8358747 0.10200099 13.180239 0.000000000
```

Here again male gender is associated with reduced time to death, while atrial fibrillation increases the hazard of death by almost four-fold. Both are statistically significant in the summary output, and we know from before that we can test a large number of other types of statistical hypotheses using the `anova` function. Again we pass `anova` the smaller (`gender_num only`) and larger (`gender_num and afib_flg`) nested models.

```
anova(gender.coxph,genderafib.coxph)
```

```
## Analysis of Deviance Table
##  Cox model: response is  datSurv
##  Model 1: ~ gender_num
##  Model 2: ~ gender_num + afib_flg
##    loglik  Chisq Df P(>|Chi|)
## 1 -3636.1
## 2 -3567.4 137.37  1 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As expected, atrial fibrillation is very statistically significant, and therefore we would like to keep it in the model.

Cox regression also allows one to use covariates which change over time. This would allow one to incorporate changes in treatment, disease severity, etc. within the same patient without need for any different methodology. The major challenge to do this is mainly in the construction of the dataset, which is discussed in some of the references at the end of this chapter. Some care is required when the time dependent covariate is only measure periodically, as the method requires that it be known at every event time for the entire cohort of patients, and not just those relevant to the patient in question. This is more practical for changes in treatment which may be recorded with some precision, particularly in a database like MIMIC II, and less so for laboratory results which may be measured at the resolution of hours, days or weeks. Interpolating between lab values or carrying the last observation forward has been shown to introduce several types of problems.

16.4.5 Caveats and Conclusions

We will conclude this brief overview of survival analysis, but acknowledge we have only scratched the surface. There are many topics we have not covered or we have only briefly touched on.

Survival analysis is distinguished from other forms of analyses covered in this Chapter, as it allows the data to be censored. As was the case for the other approaches we considered, there are modeling assumptions. For instance, it is important that the censoring is not informative of the survival time. For example, if censoring occurs when treatment is withdrawn *because* the patient is too sick to continue therapy, this would be an example of informative censoring. The validity of all methods discussed in this section are then invalid. Care should be taken to make sure you understand the censoring mechanism as to avoid any false inferences drawn.

Assessment of the proportional hazards assumption is an important part of any Cox regression analysis. We refer you to the references (particularly [17] and see `?cox.zph`) at the end of this chapter for strategies and alternatives for when the proportional hazards assumption breaks down. In some circumstances, the proportional hazards assumption is not valid, and alternative approaches can be used. As is always the case, when outcomes are dependent (e.g., one patient may contribute more than one observation), the methods discussed in this section should not be used directly. Generally the standard error estimates will be too small, and *p*-values will be incorrect. The concerns in logistic regression regarding outliers, co-linearity, missing data, and covariates with sparse outcomes apply here as well, as do the concerns about model misspecification for continuous covariates.

Survival analysis is a powerful analysis technique which is extremely relevant for health studies. We have only given a brief overview of the subject, and would encourage you to further explore these methods.

16.5 Case Study and Summary

16.5.1 Section Goals

In this section, we will work through a case study, and discuss the data analysis components which should be included in an original research article suitable for a clinical journal. We will also discuss some approaches for model and feature selection.

16.5.2 Introduction

We will now use what we learned in the previous sections to examine if indwelling arterial catheters (IAC) have any effect on patient mortality. As reiterated throughout, clearly identifying a study objective is important for a smooth data analysis. In our case, we'd like to estimate the effect of IAC on mortality, but acknowledge a few potential problem areas. First, the groups who receive IAC and and those who don't are likely different in many respects, and many of these differences likely also have some effect on mortality. Second, we would like to be able to limit ourselves on mortality events which occur in close proximity to the ICU admission. The dataset includes 28 day mortality, so that would seem to be in close proximity to the ICU admission. As for the first issue, we also have many covariates which capture some of the features we may be concerned with, including severity of illness (`sapsi_first` and `sofa_first`), age (`age`), patient gender (`gender_num`) and co-morbidities (`chf_flg`, `afib_flg`, `renal_flg`, etc.).

With all these in mind, we should have a good start on determining our study objective. In our case, it might be,

> To estimate the effect that administration of IAC during an ICU admission has on 28 day mortality in patients within the MIMIC II study who received mechanical ventilation, while adjusting for age, gender, severity of illness and comorbidities.

For now, this describes our outcome and covariates quite well. One of the first things that is often done is to describe our population by computing summary statistics of all or a subset of variables collected in the study. This description allows the reader to understand how well the study would generalize to other populations. We have made available an `R` package on GitHub that will allow one to construct preliminary forms of such a table quite quickly. To install the `R` package, first install and load the `devtools` package:

```r
install.package("devtools")
library(devtools)
```

and then install and load our package by using the `install_github` function.

```r
install_github("jraffa/MIMICbook")
library(MIMICbook);
```

Before we do any in depth analysis, let's make sure we are using the original dataset, first by removing and then reloading the `dat` data frame. In order to ensure our research is reproducible, it's a good idea to make sure the entire process of doing the analysis is documented. By starting from the original copy of the dataset, we are able to present precisely what methods we used in an analysis.

```r
rm(dat)
dat <- read.csv(url)
```

As mentioned before, recoding binary encoded variables (ones which are 0s and 1s) to the `R` data class `factor` can sometimes make interpreting the `R` output easier. The following piece of code cycles through all the columns in `dat` and converts any binary variables to a `factor`.

```r
# Identify which columns are binary coded
bincols <- colMeans((dat == 1 | dat == 0), na.rm = T) == 1
for (i in 1:length(bincols)) {
    # Turn the binary columns into a factor
    if (bincols[i]) {
        dat[[i]] <- as.factor(dat[[i]])
    }
}
```

We are now ready to generate a summary of the patient characteristics in our study. The MIMICbook package has a produce.table1 function. This generates a summary table of the data frame you pass to it, using an appropriate summary for continuous variables (average and standard deviation) and categorical variables (number and percentages) for each variable. In its most simple form, produce.table1 can be passed a data frame as an argument, which we do (passing it the dat data frame). This output is not very nice, and we can make it look nicer by using a powerful R package called knitr, which provides many tools to assist in performing reproducible research. You can find out more about knitr (which can be installed using install.packages ('knitr')), by running ?knitr on the R console after loading it. We will be using the kable command, which will take our tab1 variable—a summary table we generated using the produce.table1 function, and make it look a little nicer.

```r
tab1 <- produce.table1(dat);
library(knitr);
kable(tab1,caption = "Overall patient characteristics")
```

The row descriptors are not very informative, and what we have produced would not be usable for final publication, but it suits our purposes for now. knitr allows one to output such tables in HTML, LaTeX or even a Word document, which you can edit and make the table more informative. The results are contained in Table 16.1.

A couple things we may notice from the baseline characteristics are:

1. Some variables have a lot of missing observations (e.g., bmi, po2_first, iv_day_1).
2. None of the patients have sepsis.

Both of these points are important, and illustrates why it is always a good idea to perform basic descriptive analyses before beginning any modeling. The missing data is primarily related to weight/BMI, or lab values. For the purpose of this chapter, we are going to ignore both of these classes of variables. While we would likely want to adjust for some of these covariates in a final version of the paper, and Chap. 11 gives some useful techniques for dealing with such a situation, we are going to focus on the set of covariates we had identified in our study objective, which do not include these variables. The issue related to sepsis is also of note.

Table 16.1 Overall patient characteristics

	Average (SD), or N (%)
aline_flg==1	984 (55.4 %)
icu_los_day	3.3 (3.4)
hospital_los_day	8.1 (8.2)
age	54.4 (21.1)
gender_num==1	1025 (57.7 %) [Missing: 1]
weight_first	80.1 (22.5) [Missing: 110]
bmi	27.8 (8.2) [Missing: 466]
sapsi_first	14.1 (4.1) [Missing: 85]
sofa_first	5.8 (2.3) [Missing: 6]
service_unit==SICU	982 (55.3 %)
service_num==1	982 (55.3 %)
day_icu_intime==Saturday	278 (15.7 %)
day_icu_intime_num	4.1 (2)
hour_icu_intime	10.6 (7.9)
hosp_exp_flg==0	1532 (86.3 %)
icu_exp_flg==0	1606 (90.4 %)
day_28_flg ==0	1493 (84.1 %)
mort_day_censored	614.3 (403.1)
censor_flg==1	1279 (72 %)
sepsis_flg==0	1776 (100 %)
chf_flg==0	1563 (88 %)
afib_flg==0	1569 (88.3 %)
renal_flg==0	1716 (96.6 %)
liver_flg==0	1677 (94.4 %)
copd_flg==0	1619 (91.2 %)
cad_flg==0	1653 (93.1 %)
stroke_flg==0	1554 (87.5 %)
mal_flg==0	1520 (85.6 %)
resp_flg==0	1211 (68.2 %)
map_1st	88.2 (17.6)
hr_1st	87.9 (18.8)
temp_1st	97.8 (4.5) [Missing: 3]
spo2_1st	98.4 (5.5)
abg_count	6 (8.7)
wbc_first	12.3 (6.6) [Missing: 8]
hgb_first	12.6 (2.2) [Missing: 8]
platelet_first	246.1 (99.9) [Missing: 8]
sodium_first	139.6 (4.7) [Missing: 5]
potassium_first	4.1 (0.8) [Missing: 5]
tco2_first	24.4 (5) [Missing: 5]
chloride_first	103.8 (5.7) [Missing: 5]

(continued)

Table 16.1 (continued)

	Average (SD), or N (%)
aline_flg==1	984 (55.4 %)
bun_first	19.3 (14.4) [Missing: 5]
creatinine_first	1.1 (1.1) [Missing: 6]
po2_first	227.6 (144.9) [Missing: 186]
pco2_first	43.4 (14) [Missing: 186]
iv_day_1	1622.9 (1677.1) [Missing: 143]

Sepsis certainly would contribute to higher rates of mortality when compared to patients without sepsis, but since we do not have any patients with sepsis, we cannot and do not need to adjust for this covariate per se. What we do need to do is acknowledge this fact by revising our study objective. We originally identified our population as patients within MIMIC, but because this is a subset of MIMIC—those without sepsis, we should revise the study objective to:

> To estimate the effect that administration of IAC during an ICU admission has on 28 day mortality in patients without sepsis who received mechanical ventilation within MIMIC II, while adjusting for age, gender, severity of illness and comorbidities.

We will also *not* want to include the `sepsis_flg` variable as a covariate in any of our models, as there are no patients with sepsis within this study to estimate the effect of sepsis. Now that we have examined the basic overall characteristics of the patients, we can begin the next steps in the analysis.

The next steps will vary slightly, but it is often useful to put yourself in the shoes of a peer reviewer. What problems will a reviewer likely find with your study and how can you address them? Usually, the reviewer will want to see how the population differs for different values of the covariate of interest. In our case study, if the treated group (IAC) differed substantially from the untreated group (no IAC), then this may account for any effect we demonstrate. We can do this by summarizing the two groups in a similar fashion as was done for Table 16.1. We can reuse the `produce.table1` function, but we pass it the two groups separately by splitting the `dat` data frame into two using the `split` function (by the `aline_flg` variable), later combining them into one table using `cbind` to yield Table 16.2. It's important to ensure that the same reference groups are used across the two study groups, and that's what the labels argument is used for (see `?produce.table1` for more details).

```r
datby.aline <- split(dat, dat$aline_flg)
reftable <- produce.table1(datby.aline[[1]])
tab2 <- cbind(produce.table1(datby.aline[[1]], labels = attr(reftable, "labels")),
    produce.table1(datby.aline[[2]], labels = attr(reftable, "labels")))
colnames(tab2) <- paste0("Average (SD), or N (%)", c(", No-IAC", ", IAC"))
kable(tab2, caption = "Patient characteristics stratified by IAC administration")
```

Table 16.2 Patient characteristics stratified by IAC administration

	Average (SD), or N (%), No-IAC	Average (SD), or N (%), IAC
aline_flg==0	792 (100 %)	0 (0 %)
icu_los_day	2.1 (1.9)	4.3 (3.9)
hospital_los_day	5.4 (5.4)	10.3 (9.3)
age	53 (21.7)	55.5 (20.5)
gender_num==1	447 (56.5 %) [Missing: 1]	578 (58.7 %)
weight_first	79.2 (22.6) [Missing: 71]	80.7 (22.4) [Missing: 39]
bmi	28 (9.1) [Missing: 220]	27.7 (7.5) [Missing: 246]
sapsi_first	12.7 (3.8) [Missing: 70]	15.2 (4) [Missing: 15]
sofa_first	4.8 (2.1) [Missing: 4]	6.6 (2.2) [Missing: 2]
service_unit==MICU	480 (60.6 %)	252 (25.6 %)
service_num==0	504 (63.6 %)	290 (29.5 %)
day_icu_intime==Saturday	138 (17.4 %)	140 (14.2 %)
day_icu_intime_num	4 (2)	4.1 (2)
hour_icu_intime	9.9 (7.7)	11 .2 (8. 1)
hosp_exp_flg==0	702 (88.6 %)	830 (84.3 %)
icu_exp_flg==0	734 (92.7 %)	872 (88.6 %)
day_28_flg==0	679 (85.7 %)	814 (82.7 %)
mort_day_censored	619.1 (388.3)	610.5 (414.8)
censor_flg==1	579 (73.1 %)	700 (71.1 %)
sepsis_flg==0	792 (100 %)	984 (100 %)
chf_flg==0	695 (87.8 %)	868 (88.2 %)
afib_flg==0	710 (89.6 %)	859 (87.3 %)
renal_flg==0	764 (96.5 %)	952 (96.7 %)
liver_flg==0	754 (95.2 %)	923 (93.8 %)
copd_flg==0	711 (89.8 %)	908 (92.3 %)
cad_flg==0	741 (93.6 %)	912 (92.7 %)
stroke_flg==0	722 (91.2 %)	832 (84.6 %)
mal_flg==0	700 (88.4 %)	820 (83.3 %)
resp_flg==0	514 (64.9 %)	697 (70.8 %)
map_1st	87.5 (15.9)	88.9 (18.8)
hr_st	88.4 (18.8)	87.5 (18.7)
temp_1st	97.9 (3.8) [Missing: 3]	97.7 (5.1)
spo2_1st	98.4 (5.7)	98.5 (5.4)
abg_count	1.4 (1.6)	9.7 (10.2)
wbc_first	11.7 (6.5) [Missing: 6]	12.8 (6.6) [Missing: 2]
hgb_first	12.7 (2.2) [Missing: 6]	12.4 (2.2) [Missing: 2]
platelet_first	254.3 (104.5) [Missing: 6]	239.5 (95.6) [Missing: 2]
sodium_first	139.8 (4.8) [Missing: 3]	139.4 (4.7) [Missing: 2]
potassium_first	4.1 (0.8) [Missing: 3]	4.1 (0.8) [Missing: 2]

(continued)

Table 16.2 (continued)

	Average (SD), or N (%), No-IAC	Average (SD), or N (%), IAC
tco2_first	24.7 (4.9) [Missing: 3]	24.2 (5.1) [Missing: 2]
chloride_first	103.3 (5.4) [Missing: 3]	104.3 (5.9) [Missing: 2]
bun_first	18.9 (14.5) [Missing: 3]	19.6 (14.3) [Missing: 2]
creatinine_first	1.1 (1.2) [Missing: 4]	1.1 (1) [Missing: 2]
po2_first	223.8 (152.9) [Missing: 178]	230.1 (139.6) [Missing: 8]
pco2_first	44.9 (15.9) [Missing: 178]	42.5 (12.5) [Missing: 8]
iv_day_1	[1364.2 (1406.8) Missing: 110]	1808.4 (1825) [Missing: 33]

As you can see in Table 16.2, the IAC group differs in many respects to the non-IAC group. Patients who were given IAC tended to have higher severity of illness at baseline (`sapsi_first` and `sofa_first`), slightly older, less likely to be from the MICU, and have slightly different co-morbidity profiles when compared to the non-IAC group.

Next, we can see how the covariates are distributed among the different outcomes (death within 28 days versus alive at 28 days). This will give us an idea of which covariates may be important for affecting the outcome. The code to generate this is nearly identical to that used to produce Table 16.2, but instead, we replace `aline_flg` with `day_28_flg` (the outcome) to get Table 16.3.

```
datby.28daymort <- split(dat, dat$day_28_flg)
reftablemort <- produce.table1(datby.28daymort[[1]])
tab3 <- cbind(produce.table1(datby.28daymort[[1]], labels = attr(reftablemort,
    "labels")), produce.table1(datby.28daymort[[2]], labels = attr(reftablemort,
    "labels")))
colnames(tab3) <- paste0("Average (SD), or N (%)", c(",Alive", ",Dead"))
kable(tab3, caption = "Patient characteristics stratified by 28 day mortality")
```

As can be seen in Table 16.3, those patients who died within 28 days differ in many ways with those who did not. Those who died had higher SAPS and SOFA scores, were on average older, and had different co-morbidity profiles.

16.5.3 Logistic Regression Analysis

In Table 16.3, we see that of the 984 subjects receiving IAC, 170 (17.2 %) died within 28 days, whereas 113 of 792 (14.2 %) died in the no-IAC group. In a univariate analysis we can assess if the lower rate of mortality is statistically significant, by fitting a single covariate `aline_flg` logistic regression.

Table 16.3 Patient characteristics stratified by 28 day mortality

	Average (SD), or N (%), alive	Average (SD), or N (%), dead
aline_flg==1	814 (54.5 %)	170 (60.1 %)
icu_los_day	3.2 (3.2)	4 (4)
hospital_los_day	8.4 (8.4)	6.4 (6.4)
age	50.8 (20.1)	73.3 (15.3)
gender_num==1	886 (59.4 %) [Missing: 1]	139 (49.1 %)
weight_first	81.4 (22.7) [Missing: 77]	72.4 (19.9) [Missing: 33]
bmi	28.2 (8.3) [Missing: 392]	26 (7.2) [Missing: 74]
sapsi_first	13.6 (3.9) [Missing: 51]	17.3 (3.8) [Missing: 34]
sofa_first	5.7 (2.3) [Missing: 3]	6.6 (2.4) [Missing: 3]
service_unit==SICU	829 (55.5 %)	153 (54.1 %)
service_num==1	829 (55.5 %)	153 (54.1 %)
day_icu_intime==Saturday	235 (15.7 %)	43 (15.2 %)
day_icu_intime_num	4 (2)	4.1 (2)
hour_icu_intime	10.5 (7.9)	11 (8)
hosp_exp_flg==0	1490 (99.8 %)	42 (14.8 %)
icu_exp_flg==0	1493 (100 %)	113 (39.9 %)
day_28_flg==0	1493 (100 %)	0 (0 %)
mort_day_censored	729.6 (331.4)	6.1 (6.4)
censor_flg==1	1279 (85.7 %)	0 (0 %)
sepsis_flg==0	1493 (100 %)	283 (100 %)
chf_flg==0	1348 (90.3 %)	215 (76 %)
afib_flg==0	1372 (91.9 %)	197 (69.6 %)
renal_flg==0	1447 (96.9 %)	269 (95.1 %)
liver_flg==0	1413 (94.6 %)	264 (93.3 %)
copd_flg==0	1377 (92.2 %)	242 (85.5 %)
cad_flg==0	1403 (94 %)	250 (88.3 %)
stroke_flg==0	1386 (92.8 %)	168 (59.4 %)
mal_flg==0	1294 (86.7 %)	226 (79.9 %)
resp_flg==0	1056 (70.7 %)	155 (54.8 %)
map_1st	88.2 (17.5)	88.3 (17.9)
hr_1st	88.3 (18.4)	85.8 (20.6)
temp_1st	97.8 (4.6) [Missing: 1]	97.7 (4.5) [Missing: 2]
spo2_1st	98.6 (5)	97.8 (7.6)
abg_count	5.7 (7.7)	7.5 (12.5)
wbc_first	12.2 (6.4) [Missing: 6]	12.7 (7.5) [Missing: 2]
hgb_first	12.7 (2.2) [Missing: 6]	11.9 (2.1) [Missing: 2]

(continued)

Table 16.3 (continued)

	Average (SD), or N (%), alive	Average (SD), or N (%), dead
platelet_first	246.8 (97.3) [Missing: 6]	242.1 (112.6) [Missing: 2]
sodium_first	139.6 (4.6) [Missing: 4]	139.1 (5.4) [Missing: 1]
potassium_first	4.1 (0.8) [Missing: 4]	4.2 (0.9) [Missing: 1]
tco2_first	24.3 (4.8) [Missing: 4]	25 (5.8) [Missing: 1]
chloride_first	104.1 (5.6) [Missing: 4]	102.6 (6.4) [Missing: 1]
bun_first	18 (12.9) [Missing: 4]	26.2 (19) [Missing: 1]
creatinine_first	1.1 (1.1) [Missing: 5]	1.2 (0.9) [Missing: 1]
po2_first	231.3 (146.3) [Missing: 153]	207.9 (135.8) [Missing: 33]
pco2_first	43.3 (12.9) [Missing: 153]	43.8 (18.6) [Missing: 33]
iv_day_1	1694.2 (1709.5) [Missing: 127]	1258 (1449.4) [Missing: 16]

```
uvr.glm <- glm(day_28_flg ~ aline_flg,data=dat,family="binomial")
exp(uvr.glm$coef[-1])
```

```
## aline_flg1
##   1.254919
```

```
exp(confint(uvr.glm)[-1,]);
```

```
##     2.5 %    97.5 %
## 0.9701035 1.6285165
```

Those who received IAC had over a 25 % increase in odds of 28 day mortality when compared to those who did not receive IAC. The confidence interval includes one, so we would expect the p-value would be >0.05. Running the `summary` function, we see that this is the case.

```
##               Estimate Std. Error    z value     Pr(>|z|)
## (Intercept) -1.7932333  0.1015988 -17.650149 1.014880e-69
## aline_flg1   0.2270714  0.1320347   1.719786 8.547142e-02
```

Indeed, the p-value for `aline_flg` is about 0.09. As we saw in Table 16.2, there are likely several important covariates that differed among those who received IAC and those who did not. These may serve as confounders, and the possible association we observed in the univariate analysis may be stronger, non-existent or in the opposite direction (i.e., IAC having lower rates of mortality) depending on the situation. Our next step would be to adjust for these confounders. This is an exercise in what is known as model building, and there are several ways people do this in the literature. A common approach is to fit all univariate models (one covariate at a time, as we did with `aline_flg`, but separately for each covariate and without `aline_flg`), and perform a hypothesis test on each model. Any variables which had statistical significance under the univariate models would then be included in a multivariable model. Another approach begins with the model we just fit (`uvr.glm` which only has `aline_flg` as a covariate), and then sequentially adds variables one at a time. This approach is often called *step-wise forward selection*. We will make a choice to do *step-wise backwards selection*, which is as it sounds—the opposite direction of step-wise forward selection. Model selection is a challenging task in data analysis, and there are many other methods [18] we couldn't possibly describe in full detail here. As an overall philosophy, it is important to outline and describe the process by which you will do model selection before you actually do it and stick with the process.

In our stepwise backwards elimination procedure, we are going to fit a model containing IAC (`aline_flg`), age (`age`), gender, (`gender_num`), disease severity (`sapsi_first` and `sofa_first`), service type (`service_unit`), and comorbidities (`chf_flg`, `afib_flg`, `renal_flg`, `liver_flg`, `copd_flg`, `cad_flg`, `stroke_flg`, `mal_flg` and `resp_flg`). This is often called the *full model*, and is fit below (`mva.full.glm`). From the full model, we will proceed by eliminating one variable at a time, until we are left with a model with only statistically significant covariates. Because `aline_flg` is the covariate of interest, it will remain in the model regardless of its statistical significance. At each step we need to come up with a criteria to choose which variable we will eliminate. There are several ways of doing this, but one way we can make this decision is performing a hypothesis test for each covariate, and choosing to eliminate the covariate with the largest *p*-value, unless all *p*-values are <0.05 or the largest *p*-value is `aline_flg`, in which case we would stop or eliminate the next largest *p*-value, respectively.

Most of the covariates are binary or categorical in nature, and we've already converted them to factors. The disease severity scores (SAPS and SOFA) are continuous. We could add them as we did age, but this assumes a linear trend in the odds of death as these scores change. This may or may not be appropriate (see Fig. 16.8). Indeed, when we plot the log odds of 28 day death by SOFA score, we note that while the log odds of death generally increase as the SOFA score increases the relationship may not be linear (Fig. 16.8).

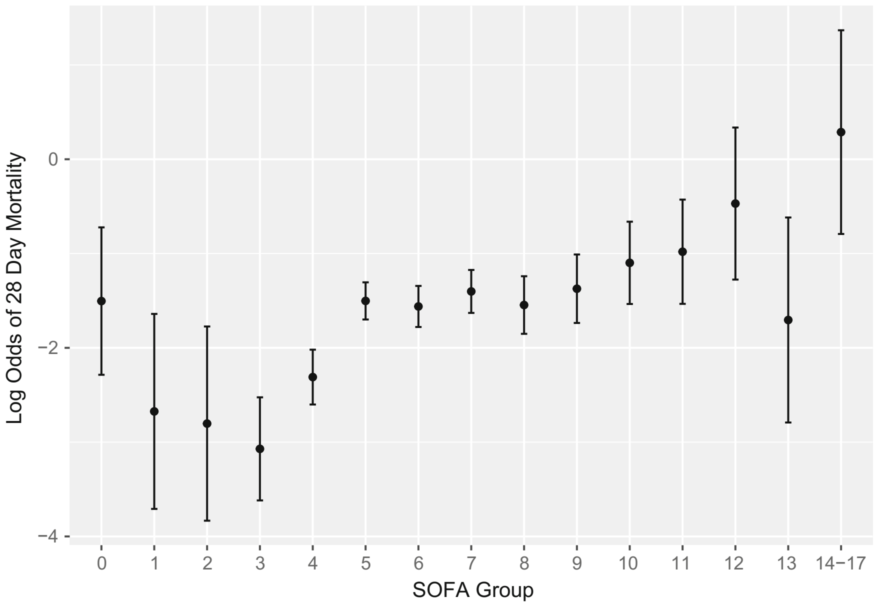


Fig. 16.8 Plot of log-odds of mortality for each of the SOFA groups. *Error bars* represent 95 % confidence intervals for the log odds

What can be done in this situation is to turn a continuous covariate into a discrete one. A quick way of doing this is using the `cut2` function in the `Hmisc` package.[2] Applying `cut2(sofa_first, g = 5)` turns the `sofa_first` variable into five approximately equal sized groups by SOFA score. For illustration, SOFA breaks down into the following sized groups by SOFA scores:

```
library(Hmisc)
table(cut2(dat$sofa_first,g=5))
```

```
##
## [0, 5)      5      6 [7, 9) [9,17]
##    523    346    294    391    216
```

with not quite equal groups, due to the already discretized nature of SOFA to begin with. We will treat both SAPS and SOFA in this way in order to avoid any model misspecification that may occur as a result of assuming a linear relationship.

Returning to fitting the full model, we use these new disease severity scores, along with the other covariates we identified to include in the full model.

[2] You may need to install `Hmisc`, which can be done by running `install.packages('Hmisc')` from the `R` command prompt.

```
mva.full.glm <- glm(day_28_flg ~ aline_flg + age + gender_num + cut2(sapsi_first,
    g = 5) + cut2(sofa_first, g = 5) + service_unit + chf_flg + afib_flg + renal_flg +
    liver_flg + copd_flg + cad_flg + stroke_flg + mal_flg + resp_flg, data = dat,
    family = "binomial")
summary(mva.full.glm)
```

```
##
## Call:
## glm(formula = day_28_flg ~ aline_flg + age + gender_num + cut2(sapsi_first,
##     g = 5) + cut2(sofa_first, g = 5) + service_unit + chf_flg +
##     afib_flg + renal_flg + liver_flg + copd_flg + cad_flg + stroke_flg +
##     mal_flg + resp_flg, family = "binomial", data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2912  -0.4710  -0.2330  -0.1104   2.9640
##
## Coefficients:
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       -7.61471    0.86262  -8.827  < 2e-16 ***
## aline_flg1                         0.01085    0.20443   0.053 0.957679
## age                                0.04020    0.00627   6.412 1.44e-10 ***
## gender_num1                        0.16214    0.17296   0.937 0.348527
## cut2(sapsi_first, g = 5)[12,14)    0.36961    0.40348   0.916 0.359637
## cut2(sapsi_first, g = 5)[14,16)    1.01794    0.36214   2.811 0.004940 **
## cut2(sapsi_first, g = 5)[16,19)    0.92803    0.36794   2.522 0.011662 *
## cut2(sapsi_first, g = 5)[19,32]    1.77615    0.37446   4.743 2.10e-06 ***
## cut2(sofa_first, g = 5)5           0.49761    0.30267   1.644 0.100159
## cut2(sofa_first, g = 5)6           0.58530    0.30300   1.932 0.053396 .
## cut2(sofa_first, g = 5)[7, 9)      0.68011    0.29439   2.310 0.020876 *
## cut2(sofa_first, g = 5)[9,17]      0.75134    0.34062   2.206 0.027397 *
## service_unitMICU                   1.08086    0.67839   1.593 0.111100
## service_unitSICU                   0.64257    0.67144   0.957 0.338562
## chf_flg1                           0.23350    0.23381   0.999 0.317962
## afib_flg1                          0.52408    0.21122   2.481 0.013092 *
## renal_flg1                        -0.76796    0.40904  -1.877 0.060452 .
## liver_flg1                         0.47238    0.34032   1.388 0.165125
## copd_flg1                          0.23440    0.24631   0.952 0.341287
## cad_flg1                          -0.25674    0.28823  -0.891 0.373065
## stroke_flg1                        2.04301    0.21966   9.301  < 2e-16 ***
## mal_flg1                           0.49319    0.20897   2.360 0.018274 *
## resp_flg1                          0.69330    0.19166   3.617 0.000298 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1400.58  on 1683  degrees of freedom
## Residual deviance:  954.39  on 1661  degrees of freedom
##   (92 observations deleted due to missingness)
## AIC: 1000.4
##
## Number of Fisher Scoring iterations: 6
```

The `summary` output show that some of the covariates are very statistically significant, while others may be expendable. Ideally, we would like as simple of a model as possible that can explain as much of the variation in the outcome as possible. We will attempt to remove our first covariate by the procedure we outlined above. For each of the variables we consider removing, we could fit a logistic regression model without that covariate, and then test it against the current model. R has a useful function that automates this process for us, called drop1. We pass to drop1 our logistic regression object (mva.full.glm) and the type of test you would like to do. If you recall from the logistic regression section, we used test = ”Chisq”, and this is what we will pass the drop1 function as well.

```
drop1(mva.full.glm,test="Chisq")
```

```
## Single term deletions
##
## Model:
## day_28_flg ~ aline_flg + age + gender_num + cut2(sapsi_first,
##     g = 5) + cut2(sofa_first, g = 5) + service_unit + chf_flg +
##     afib_flg + renal_flg + liver_flg + copd_flg + cad_flg + stroke_flg +
##     mal_flg + resp_flg
##                           Df Deviance     AIC    LRT  Pr(>Chi)
## <none>                         954.39 1000.39
## aline_flg                  1   954.39  998.39  0.003 0.9576771
## age                        1  1000.60 1044.60 46.210 1.063e-11 ***
## gender_num                 1   955.27  999.27  0.883 0.3475044
## cut2(sapsi_first, g = 5)   4   989.69 1027.69 35.304 4.023e-07 ***
## cut2(sofa_first, g = 5)    4   960.95  998.95  6.558 0.1611514
## service_unit               2   960.11 1002.11  5.716 0.0573820 .
## chf_flg                    1   955.38  999.38  0.990 0.3196816
## afib_flg                   1   960.47 1004.47  6.080 0.0136708 *
## renal_flg                  1   958.20 1002.20  3.814 0.0508182 .
## liver_flg                  1   956.23 1000.23  1.839 0.1750410
## copd_flg                   1   955.28  999.28  0.893 0.3445691
## cad_flg                    1   955.20  999.20  0.811 0.3678829
## stroke_flg                 1  1045.22 1089.22 90.831 < 2.2e-16 ***
## mal_flg                    1   959.80 1003.80  5.410 0.0200201 *
## resp_flg                   1   967.57 1011.57 13.177 0.0002834 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As you see from the output, each covariate is listed, along with a *p*-value (Pr (> Chi)). Each row represents a hypothesis test with the bigger (alternative model) being the full model (mva.full.glm), and each null being the full model without the row’s covariate. The *p*-values here should match those output if you were to do this exact test with anova. As we can see from the listed *p*-values, aline_flg has the largest *p*-value, but we stipulated in our model selection plan that we would retain this covariate as it’s our covariate of interest. We will then go to the next largest *p*-value which is the cad_flg variable (coronary artery disease). We will update our model, and repeat the backwards elimination step on the updated model. We could just cut and paste the mva.full.glm command and remove + cad_flg, but an easier way less prone to errors is to use the update command. The update function can take a glm or lm object, and alter one of the covariates. To do a backwards elimination, the second argument is.∼. - variable. The .∼. part indicates keep the outcome and the rest of the variables the same, and the - variable indicates to fit the model without the variable called variable. Hence, to fit a new model from the full model, but without the cad_flg variable, we would run:

```
mva.tmp.glm <- update(mva.full.glm, .~. - cad_flg)
```

We then repeat the drop1 step:

```
drop1(mva.tmp.glm,test="Chisq")
```

```
## Single term deletions
##
## Model:
## day_28_flg ~ aline_flg + age + gender_num + cut2(sapsi_first,
##     g = 5) + cut2(sofa_first, g = 5) + service_unit + chf_flg +
##     afib_flg + renal_flg + liver_flg + copd_flg + stroke_flg +
##     mal_flg + resp_flg
##                          Df Deviance     AIC    LRT  Pr(>Chi)
## <none>                       955.20  999.20
## aline_flg                 1   955.20  997.20  0.002 0.9674503
## age                       1  1000.92 1042.92 45.715 1.368e-11 ***
## gender_num                1   955.98  997.98  0.784 0.3760520
## cut2(sapsi_first, g = 5)  4   990.38 1026.38 35.180 4.266e-07 ***
## cut2(sofa_first, g = 5)   4   961.75  997.75  6.552 0.1615399
## service_unit              2   960.98 1000.98  5.782 0.0555160 .
## chf_flg                   1   955.92  997.92  0.719 0.3965762
## afib_flg                  1   961.32 1003.32  6.115 0.0134006 *
## renal_flg                 1   959.97 1001.97  4.774 0.0288966 *
## liver_flg                 1   957.06  999.06  1.862 0.1723427
## copd_flg                  1   956.02  998.02  0.824 0.3640764
## stroke_flg                1  1045.73 1087.73 90.526 < 2.2e-16 ***
## mal_flg                   1   960.64 1002.64  5.435 0.0197326 *
## resp_flg                  1   968.84 1010.84 13.638 0.0002217 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

and see that aline_flg still has the largest p-value, but chf_flag has the second largest, so we'll choose to remove it next. To update the new model, and run another elimination step, we would run:

```
mva.tmp.glm2 <- update(mva.tmp.glm, .~. - chf_flg)
drop1(mva.tmp.glm2,test="Chisq")
```

```
## Single term deletions
##
## Model:
## day_28_flg ~ aline_flg + age + gender_num + cut2(sapsi_first,
##     g = 5) + cut2(sofa_first, g = 5) + service_unit + afib_flg +
##     renal_flg + liver_flg + copd_flg + stroke_flg + mal_flg +
##     resp_flg
##                          Df Deviance     AIC    LRT  Pr(>Chi)
## <none>                        955.92  997.92
## aline_flg                 1   955.93  995.93  0.016 0.9003547
## age                       1  1005.90 1045.90 49.976 1.556e-12 ***
## gender_num                1   956.65  996.65  0.734 0.3916088
## cut2(sapsi_first, g = 5)  4   991.04 1025.04 35.121 4.387e-07 ***
## cut2(sofa_first, g = 5)   4   962.39  996.39  6.467 0.1669071
## service_unit              2   962.45 1000.45  6.529 0.0382253 *
## afib_flg                  1   963.01 1003.01  7.090 0.0077512 **
## renal_flg                 1   960.24 1000.24  4.321 0.0376445 *
## liver_flg                 1   957.70  997.70  1.780 0.1821692
## copd_flg                  1   956.95  996.95  1.035 0.3088774
## stroke_flg                1  1045.73 1085.73 89.808 < 2.2e-16 ***
## mal_flg                   1   961.15 1001.15  5.231 0.0221921 *
## resp_flg                  1   970.13 1010.13 14.214 0.0001632 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

where again aline_flg has the largest *p*-value, and gender_num has the second largest. We continue, eliminating gender_num, copd_flg, liver_flg, cut2(sofa_first, g = 5), renal_flg, and service_unit, in that order (results omitted). The table produced by drop1 from our final model is as follows:

```
drop1(mva.tmp.glm8,test="Chisq")
```

```
## Single term deletions
##
## Model:
## day_28_flg ~ aline_flg + age + cut2(sapsi_first, g = 5) + afib_flg +
##     stroke_flg + mal_flg + resp_flg
##                          Df Deviance    AIC    LRT  Pr(>Chi)
## <none>                        989.10 1011.1
## aline_flg                 1   989.10 1009.1  0.001  0.977380
## age                       1  1037.65 1057.7 48.556 3.209e-12 ***
## cut2(sapsi_first, g = 5)  4  1037.88 1051.9 48.788 6.465e-10 ***
## afib_flg                  1   995.60 1015.6  6.502  0.010777 *
## stroke_flg                1  1078.58 1098.6 89.485 < 2.2e-16 ***
## mal_flg                   1   997.37 1017.4  8.274  0.004021 **
## resp_flg                  1  1022.30 1042.3 33.200 8.317e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

All variables are statistically significant at the 0.05 significance level. Looking at the summary output, we see that aline_flg is not statistically significant (p = 0.98), but all other terms are statistically significant, with the exception of the cut2(sapsi_first, g = 5)[12,14), which suggest that the second to lowest SAPS group may not be statistically significantly different than the baseline (lowest SAPS group).

```
mva.final.glm <- mva.tmp.glm8;
summary(mva.final.glm)
```

```
##
## Call:
## glm(formula = day_28_flg ~ aline_flg + age + cut2(sapsi_first,
##     g = 5) + afib_flg + stroke_flg + mal_flg + resp_flg, family = "binomial",
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3025  -0.4928  -0.2433  -0.1289   3.1103
##
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     -6.081944   0.445625 -13.648  < 2e-16 ***
## aline_flg1                       0.005078   0.179090   0.028  0.97738
## age                              0.037205   0.005644   6.592 4.33e-11 ***
## cut2(sapsi_first, g = 5)[12,14)  0.302084   0.391502   0.772  0.44035
## cut2(sapsi_first, g = 5)[14,16)  1.127302   0.344670   3.271  0.00107 **
## cut2(sapsi_first, g = 5)[16,19)  1.030901   0.347842   2.964  0.00304 **
## cut2(sapsi_first, g = 5)[19,32]  1.883738   0.347311   5.424 5.84e-08 ***
## afib_flg1                        0.522664   0.203485   2.569  0.01021 *
## stroke_flg1                      1.870553   0.199980   9.354  < 2e-16 ***
## mal_flg1                         0.592458   0.202297   2.929  0.00340 **
## resp_flg1                        0.976808   0.171629   5.691 1.26e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1413.4  on 1690  degrees of freedom
## Residual deviance:  989.1  on 1680  degrees of freedom
##   (85 observations deleted due to missingness)
## AIC: 1011.1
##
## Number of Fisher Scoring iterations: 6
```

We would call this model our final model, and would present it in a table similar to Table 16.4. Since the effect of IAC was of particular focus, we will highlight it by saying that it is not associated with 28 day mortality with an estimated adjusted odds ratio of 1.01 (95 % CI: 0.71–1.43, p = 0.98). We may conclude that after adjusting for the other potential confounders found in Table 16.4, we do not find any statistically significant impact of using IAC on mortality.

16.5.4 Conclusion and Summary

This brief overview of the modeling techniques for health data has provided you with the foundation to perform the most common types of analyses in health studies. We have cited how important having a clear study objective before

Table 16.4 Multivariable logistic regression analysis for mortality at 28 days outcome (final model

Covariate	AOR	Lower 95 % CI	Upper 95 % CI	*p*-value
IAC	1.01	0.71	1.43	0.977
Age (per year increase)	1.04	1.03	1.05	<0.001
SAPSI [12–14)* (relative to SAPSI <2)	1.35	0.63	2.97	0.440
SAPSI [14–16)*	3.09	1.61	6.28	0.001
SAPSI [16–19)*	2.80	1.45	5.74	0.003
SAPSI [19–32]*	6.58	3.42	13.46	<0.001
Atrial fibrillation	1.69	1.13	2.51	0.010
Stroke	6.49	4.40	9.64	<0.001
Malignancy	1.81	1.21	2.68	0.003
Non-COPD respiratory disease	2.66	1.90	3.73	<0.001

conducting data analysis is, as it identifies all the important aspects you need to plan and execute your analysis. In particular by identifying the outcome, you should be able to determine what analysis methodology would be most appropriate. Often you will find that you will be using multiple analysis techniques for different study objectives within the same study. Table 16.5 summarizes some of the important aspects of each analysis approach.

Fortunately, R's framework for conducting these analyses is very similar across the different types of techniques, and this framework will often extend more generally to other more complex models (including machine learning algorithms) and data structures (including dependent/correlated data such as longitudinal data).

Table 16.5 Summary of different methods

	Linear regression	Logistic regression	Cox proportional hazards model
Outcome data type	Continuous	Binary	Time to an event (possibly censored)
Useful preliminary analysis	Scatterplot	Contingency and 2 × 2 tables	Kaplan-Meier survivor function estimate
Presentation Output	Coefficient	Odds Ratio	Hazard ratio
R output	Coefficient	Log Odds ratio	Log hazard ratio
Presentation Interpretation	An estimate of the expected change in the outcome per one unit increase in the covariate, while keeping all other covariates constant	An estimate of the fold change in the odds of the outcome per unit increase in the covariate, while keeping all other covariates constant	An estimate of the fold change in the hazards of the outcome per unit increase in the covariate, while keeping all other covariates constant

We have highlighted some areas of concern that careful attention should be paid to including missing data, colinearity, model misspecification, and outliers. Some of these items will be looked at more closely in Chap. 17.

References

1. Hsu DJ, Feng M, Kothari R, Zhou H, Chen KP, Celi LA (2015) The association between indwelling arterial catheters and mortality in hemodynamically stable patients with respiratory failure: a propensity score analysis. CHEST J 148(6):1470–1476
2. Goldberger AL, Amaral LA, Glass L, Hausdorff JM, Ivanov PC, Mark RG, Mietus JE, Moody GB, Peng C-K, Stanley HE (2000) Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. Circulation 101(23):e215–e220
3. Indwelling arterial catheter clinical data from the MIMIC II database (2016) Available http://physionet.org/physiobank/database/mimic2-iaccd/. Accessed: 02 Jun 2016
4. Friedman J, Hastie T, Tibshirani R (2009) The elements of statistical learning: data mining, inference, and prediction. Springer series in statistics
5. James G, Witten D, Hastie T, Tibshirani R (2013) An introduction to statistical learning, vol 112. Springer, Berlin
6. Harrell F (2015) Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis. Springer, Berlin
7. Venables WN, Ripley BD (2013) Modern applied statistics with S-plus. Springer Science & Business Media
8. Weisberg S (2005) Applied linear regression, vol 528. Wiley, New York
9. Diggle P, Heagerty P, Liang KY, Zeger S (2013) Analysis of longitudinal data. OUP Oxford
10. McCullagh P, Nelder JA (1989) Generalized linear models, vol 37. CRC press, Boca Raton
11. Hosmer DW, Lemeshow S (2004) Applied logistic regression. Wiley, New York
12. Kleinbaum DG, Klein M (2006) Survival analysis: a self-learning text. Springer Science & Business Media
13. Kaplan EL, Meier P (1958) Nonparametric estimation from incomplete observations. J Am Stat Assoc 53(282):457–481
14. Cox DR (1972) Regression models and life-tables. J R Stat Soc Ser B (Methodol) 34(2):187–220
15. Collett D (2015) Modelling survival data in medical research. CRC press, Boca Raton
16. Kalbfleisch JD, Prentice RL (2011) The statistical analysis of failure time data, vol 360. Wiley, New York
17. Therneau TM, Grambsch PM (2000) Modeling survival data: extending the Cox model. Springer Science & Business Media
18. Dash M, Liu H (1997) Feature selection for classification. Intel Data Anal 1(3):131–156

Chapter 17
Sensitivity Analysis and Model Validation

Justin D. Salciccioli, Yves Crutain, Matthieu Komorowski and Dominic C. Marshall

Learning Objectives

- Appreciate that all models possess inherent limitations for generalizability.
- Understand the assumptions for making causal inferences from available data.
- Check model fit and performance.

17.1 Introduction

Imagine that you have now finished the primary analyses of your current research and have been able to reject the null hypothesis. Even after your chosen methods have been applied and robust models generated, some doubts may remain. "*How confident are you in the results? How much will the results change if your basic data is slightly wrong? Will that have a minor impact on your results? Or will it give a completely different outcome?*" Causal inference is often limited by the assumptions made in study design and analysis and this is particularly pronounced when working with observational health data. An important approach for any investigator is to avoid relying on any single analytical approach to assess the hypothesis and as such, a critical next step is to test the assumptions made in the analysis.

Sensitivity Analysis and Model Validation are linked in that they are both attempts to assess the appropriateness of a particular model specification and to appreciate the strength of the conclusions being drawn from such a model. Whereas model validation is useful for assessing the model fit within a specific research dataset, sensitivity analysis is particularly useful in gaining confidence in the results of the primary analysis and is important in situations where a model is likely to be used in a future research investigation or in clinical practice. Herein, we discuss

MIT Critical Data, *Secondary Analysis of Electronic Health Records*,
DOI 10.1007/978-3-319-43742-2_17

concepts relating to the assessment of model fit and outline broadly the steps relating to cross and external validation with direct application to the arterial line project. We will discuss briefly a few of the common reasons why models fail validity testing and the potential implications of such failure.

17.2 Part 1—Theoretical Concepts

17.2.1 Bias and Variance

In statistics and machine learning, the bias–variance trade-off (or dilemma) is the problem of simultaneously minimizing two sources of error that prevent supervised learning algorithms from generalizing beyond their training set. A model with high bias fails to accurately estimate the data. For example, a linear regression model would have high bias when trying to model a quadratic relationship—no matter how the parameters are set (as shown in Fig. 17.1). Variance, on the other hand, relates to the stability of your model in response to new training examples. An algorithm that fits the training data very well but generalizes poorly to new examples (showing over-fitting) is said to have high variance.

Some common strategies for dealing with bias and variance are outlined below.

- High bias:
  - Adding features (predictors) tends to decrease bias, at the expense of introducing additional variance.
  - Adding training examples will not fix high bias, because the underlying model will still not be able to approximate the correct function.
- High variance:
  - Reducing model complexity can help decrease variance. Dimensionality reduction and feature selection are two examples of methods to decrease model parameters and thus reduce variance (parameter selection is discussed below).
  - A larger training set tends to decrease variance.

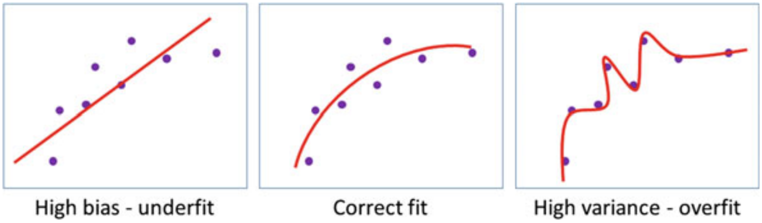


Fig. 17.1 Comparison between bias and variance in model development

17.2.2 Common Evaluation Tools

A variety of statistical techniques exist to quantitatively assess the performance of statistical models. These techniques are important, but generally beyond the scope of this textbook. We will, however, briefly mention two of the most common techniques: the R^2 value used for regressions and the Receiver Operating Characteristic (ROC) curve used for binary classifier (dichotomous outcome).

The R^2 value is a summary statistic representing the proportion of total variance in the outcome variable that is captured by the model. The R^2 has a range from 0 to 1 where values close to 0 reflect situations where the model does not appreciably summarise variation in the outcome of interest and values close to 1 indicate that the model captures nearly all of the variation in the outcome of interest. High R^2 values means that a high proportion of the variance is explained by the regression model. In R programming, the R^2 is computed when the linear regression function is used. For an example of R-code to produce the R^2 value please refer to the "R^{2}" function.

The R^2 value is an overall measure of strength of association between the model and the outcome and does not reflect the contribution of any single independent predictor variable. Further, while we may expect intuitively that there is a proportional relationship between the number of predictor variables and the overall model R^2, in practice, adding predictors does not necessarily increase R^2 in new data. It is possible for an individual predictor to decrease the R^2 depending on how this variable interacts with the other parameters in the model.

For the purpose of this discussion we expect the reader to be familiar with the computation and utility of the values of sensitivity and specificity. In situations such as developing a new diagnostic test, investigators may define a single threshold value to classify a test result as positive. When dealing with a dichotomous outcome, the Receiver Operating Characteristic (ROC) curve is a more complete description of a model's ability to classify outcomes. The ROC curve is a common method to show the relationship between the sensitivity of a classification model and its false positive rate (1 - specificity). The resultant Area Under the Curve of the ROC reflects the prediction estimate of the model, can take values from 0.5 to 1 with values of 0.5 implying near random chance in outcomes and values nearer to 1 reflecting greater prediction. For an example of ROC curves in R, please refer to the "ROC" function in the accompanying code. For further reading on the ROC curve, see for example the article by Fawcett [1] (Fig. 17.2).

17.2.3 Sensitivity Analysis

Sensitivity analysis involves a series of methods to quantify how the uncertainty in the output of a model is related to the uncertainty in its inputs. In other words, sensitivity analysis assesses how "sensitive" the model is to fluctuations in the parameters and data on which it is built. The results of sensitivity analysis can have

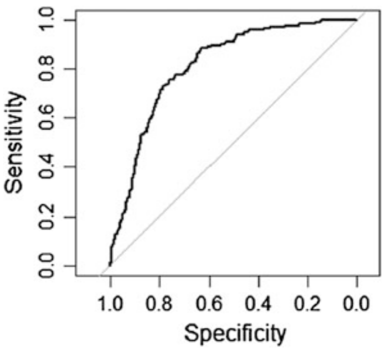


Fig. 17.2 Example of receiver operator characteristic (ROC) curve which may be used to assess the ability of a model to discriminate between dichotomous outcomes

important implications at many stages of the modeling process, including for identifying errors in the model itself, informing the calibration of model parameters, and exploring more broadly the relationship between the inputs and outputs of the model.

The principles of a sensitivity analysis are: (a) to allow the investigator to quantify the uncertainty in a model, (b) to test the model of interest using a secondary experimental design, and (c) using the results of the secondary experimental design to calculate the overall sensitivity of the model of interest. The justification for sensitivity analysis is that a model will always perform better (i.e. over-perform) when tested on the dataset from which it was derived. Sub-group analysis is a common variation of sensitivity analysis [2].

17.2.4 Validation

As discussed in Chap. 16—Data Analysis validation is used to confirm that the model of interest will perform similarly under modified testing conditions. As such, it is the primary responsibility of the investigator to assess the suitability of model fit to the data. This may be accomplished with a variety of methodological approaches and for a more detailed discussion of model fit diagnostics the reader is referred to other sources [3]. Although it is beyond the scope of this textbook to discuss validation in detail, the general theory is to select a model based on two principles: model parsimony and clinical relevance. A number of pre-defined model selection algorithm-based approaches including Forward selection, Backward, and Stepwise selection, but also lasso and genetic algorithms, available in common statistical packages. Please refer to Chap. 16 for further information about model selection.

Cross validation is a technique used to assess the predictive ability of a regression model. The approach has been discussed in detail previously [4]. The concept of cross-validation relies on the principle that a large enough dataset can split into two or more (not necessarily equally sized) sub-groups, the first being used to derive the model and the additional data set(s) reserved for model testing and validation. To avoid losing information by training the model only on a subset of available data, a variant called k-fold cross validation exist (not discussed here).

External validation is defined as testing the model on a sample of subjects taken from a population different than the original cohort. External validation is usually a more robust approach for testing the derived model in that the maximum amount of information has been used from the initial dataset to derive a model and an entirely independent dataset is used subsequently to verify the suitability of the model of interest. Although external validation is the most rigorous and an essential validation method, finding a suitably similar albeit entirely independent cohort for external validation is challenging and is often unavailable for researchers. However, with the increasing amount of healthcare data being captured electronically it is likely that researchers will also have increasing capacity for external validation.

17.3 Case Study: Examples of Validation and Sensitivity Analysis

This case study used the dataset produced for the "IAC study", which evaluated the impact of inserting an arterial line in intensive care patients with respiratory failure. Three different sensitivity analyses were performed:

1. Test the effects of varying the inclusion criteria of time to mechanical ventilation and mortality;
2. Test the effects of changes in caliper level for propensity matching on association between arterial catheter insertion and the mortality;
3. Hosmer-Lemeshow Goodness-of-Fit test to assess the overall fit of the data to the model of interest.

A number of R packages from CRAN, were used to conduct these analyses: Multivariate and Propensity Score Matching [5], analysis of complex survey samples [6], ggplot2 for generating graphics [7], pROC for ROC curves [8] and Twang for weighting and analyzing non-equivalent groups [9].

17.3.1 Analysis 1: Varying the Inclusion Criteria of Time to Mechanical Ventilation

The first sensitivity analysis evaluates the effect of varying the inclusion criteria of time to mechanical ventilation and mortality. Mechanical ventilation is one of the more common invasive interventions performed in the ICU and the timing of intervention may serve as a surrogate for the severity of critical illness, as we might

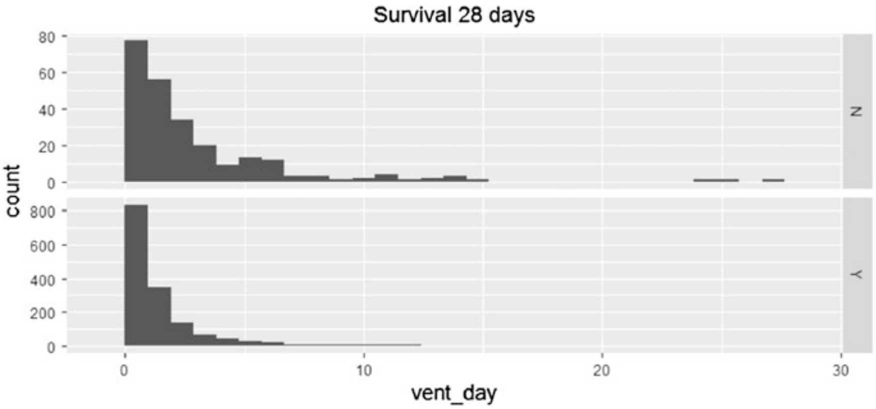


Fig. 17.3 Simple sensitivity analysis to compare outcomes between groups by varying the inclusion criteria. Modification of the inclusion criteria for subjects entered into the model is a common sensitivity analysis

expect patients with worse illness to require assisted ventilation earlier in the course of intensive care. As such, mechanical ventilation along with indwelling arterial catheter (IAC), another invasive intervention, may both be related to the outcome of interest, 28-day mortality. An example of R-code to inspect the distribution across groups of patients by ventilation status is provided in the "Cohort" function, in the accompanying R functions document (Fig. 17.3).

By modifying the time of first assisted mechanical ventilation we may also obtain important information about the effect of the primary exposure on the outcome. An example of R-code for this analysis is provided in the "Ventilation" function.

17.3.2 Analysis 2: Changing the Caliper Level for Propensity Matching

The second sensitivity analysis performed tests the impact of different caliper levels for propensity matching on the association between arterial catheter and the mortality. In this study, the propensity score matches a subject who did not received an arterial catheter with a subject who did. The matching algorithm creates a pair of two independent subjects whose propensity scores are the most similar. However, the investigator is responsible for setting a maximum reasonable difference in propensity score which would allow the matching algorithm to generate a suitable match; this maximum reasonable difference is also known as the propensity score 'caliper'. The choice of caliper for the propensity score match will directly influence the variance bias trade-off such that a wider caliper will result in matching of subjects which are more dissimilar with respect to likelihood of treatment. An

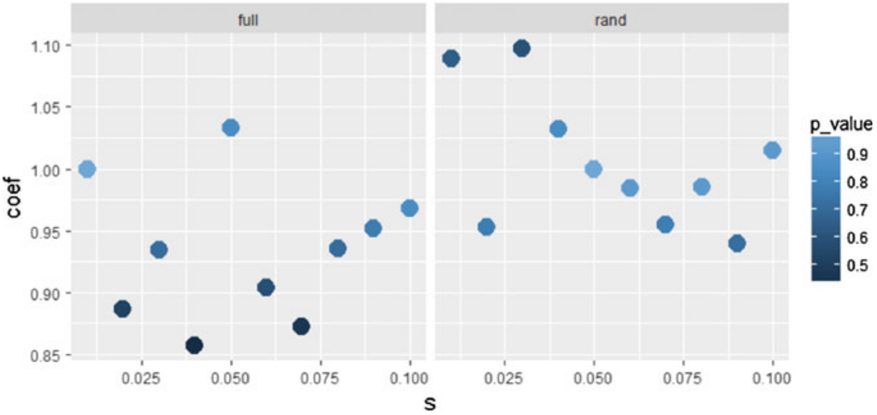


Fig. 17.4 A sensitivity analysis to assess the effect of modifying the propensity score caliper level

example of the R-code to produce a sensitivity analysis for varying the propensity score caliper level is provided in the accompanying R functions document as the "Caliper" function.

The Fig. 17.4 displays the effect of adjustments of the caliper level on the propensity score. The full model shows a lower coefficient due to the presence of additional variables.

17.3.3 Analysis 3: Hosmer-Lemeshow Test

The Hosmer-Lemeshow Goodness-of-Fit test may be used to assess the overall fit of the data to the model of interest [10]. For this test, the subjects are grouped according to a percentile of risk (usually deciles). A Pearson Chi square statistic is generated to compare observed subject grouping with the expected risk according to the model. An example of the R-code to conduct this test is provided in the accompanying R functions document as the "HL" function.

17.3.4 Implications for a 'Failing' Model

In the favorable situation of a robust model, each sensitivity analysis and validation technique supports the model as an appropriate summary of the data. However, in some situations, the chosen validation method or sensitivity analysis reveals an inadequate fit of the model for the data such that the model fails to accurately predict the outcome of interest. A 'failing' model may be the result of a number of different factors. Occasionally, it is possible to modify the model derivation procedure in order to claim a better fit on the data. In the situations where modifying the model does not allow to achieve an acceptable level of error, however, it is good practice to renounce the investigation and re-start with an assessment of the a priori assumptions, in an attempt to develop a different model.

17.4 Conclusion

The analysis of observational health data carries the inherent limitation of unmeasured confounding. After model development and primary analysis, an important step is to confirm a model's performance with a series of confirmatory tests to verify a valid model. While validation may be used to check that the model is an appropriate fit for the data and is likely to perform similarly in other cohorts, sensitivity analysis may be used to interrogate inherent assumptions of the primary analysis. When performed adequately these additional steps help improve the robustness of the overall analysis and aid the investigator in making meaningful inferences from observational health data.

Take Home Messages

1. Validation and sensitivity analyses test the robustness of the model assumptions and are a key step in the modeling process;
2. The key principle of these analyses is to vary the model assumptions and observe how the model responds;
3. Failing the validation and sensitivity analyses might require the researcher to start with a new model.

Code Appendix

The code used in this chapter is available in the GitHub repository for this book: https://github.com/MIT-LCP/critical-data-book. Further information on the code is available from this website.

References

1. Fawcett T (2006) An introduction to ROC analysis. Pattern Recogn Lett 27(8):861–874
2. Brookes ST, Whitely E, Egger M, Smith GD, Mulheran PA, Peters TJ (2004) Subgroup analyses in randomized trials: risks of subgroup-specific analyses; power and sample size for the interaction test. J Clin Epidemiol 57(3):229–236
3. Pregibon D (1981) Logistic regression diagnostics. Ann Stat 9(4):705–724
4. Picard RR, Cook RD (1984) Cross-validation of regression models. J Am Stat Assoc 79 (387):575–583
5. Sekhon JS (2011) Multivariate and propensity score matching software with automated balance optimization: the matching package for R. J Stat Softw 42(i07)
6. Lumley T (2004) Analysis of complex survey samples. J Stat Softw 09(i08)
7. Wickham H (2009) ggplot2. Springer, New York
8. Robin X, Turck N, Hainard A, Tiberti N, Lisacek F, Sanchez J-C, Müller M (2011) pROC: an open-source package for R and S + to analyze and compare ROC curves. BMC Bioinf 12:77
9. Ridgeway G, Mccaffrey D, Morral A, Burgette L, Griffin BA (2006) Twang: toolkit for weighting and analysis of nonequivalent groups. R package version 1.4-9.3. In: R Foundation for Statistical Computing, 2006. (http://www.cran.r-project.org). Accessed 2015
10. Hosmer DW, Lemesbow S (1980) Goodness of fit tests for the multiple logistic regression model. Commun Stat Theory Methods 9(10):1043–1069

Part III
Case Studies Using MIMIC

Introduction

This section presents twelve case studies of secondary analyses of electronic health records (EHRs). The case studies exhibit a wide range of research topics and methodologies, making them of interest to a wide range of researchers. They are written primarily for the beginner, although the experienced researcher will also benefit much from the detailed explanations offered by experts in the field. The case studies provide an opportunity to thoroughly engage with high-level research studies, since they are accompanied by both publicly available data and analytical code. This section should not be approached as a continuous narrative. Rather, each case study can be read independently. Indeed, it is advisable to begin with those which lie closest to your interests. An overview of the research areas and methodologies of the case studies is now provided.

The case studies are ordered according to their research areas. The first two case studies concern system-level analyses, beginning with an analysis of the trends in clinical practice with regard to mechanical ventilation (Chap. 18). This is followed by an investigation into the effect of caring for critically-ill patients in "non-target ICUs", otherwise known as boarding, on mortality (Chap. 19). The next three case studies focus on mortality prediction using a plethora of inputs such as demographics, vital signs and laboratory test results (Chaps. 20–22). Two case studies investigate the effectiveness of a clinical intervention, with assessments of clinical effectiveness (Chap. 23) and cost effectiveness (Chap. 24). A study of the relationship between blood pressure and the risk of Acute Kidney Injury is presented, illustrating the physiological insights that can be gained by analysis of EHRs (Chap. 25). Two case studies are then presented on monitoring techniques: an investigation into the estimation of respiratory rate, a key physiological parameter, from routinely acquired physiological signals (Chap. 26); and a detailed study of the potential for false alarm reduction using machine learning classification techniques (Chap. 27). Finally two studies consider particular aspects of research methodology, focusing on patient cohort identification (Chap. 28) and mathematical techniques for selection of hyperparameters (Chap. 29).